AWS Security Best Practices

August 2016

Notices

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

Contents

Introduction 1
Know the AWS Shared Responsibility Model 2
Understanding the AWS Secure Global Infrastructure 3
Sharing Security Responsibility for AWS Services 4
Using the Trusted Advisor Tool 10
Define and Categorize Assets on AWS 10
Design Your ISMS to Protect Your Assets on AWS 11
Manage AWS Accounts, IAM Users, Groups, and Roles 13
Strategies for Using Multiple AWS Accounts 14
Managing IAM Users 15
Managing IAM Groups 15
Managing AWS Credentials 16
Understanding Delegation Using IAM Roles and Temporary Security Credentials 17
Managing OS-level Access to Amazon EC2 Instances 20
Secure Your Data 22
Resource Access Authorization 22
Storing and Managing Encryption Keys in the Cloud 23
Protecting Data at Rest 24
Decommission Data and Media Securely 31
Protect Data in Transit 32
Secure Your Operating Systems and Applications 38
Creating Custom AMIs 39
Bootstrapping 41
Managing Patches 42
Controlling Security for Public AMIs 42
Protecting Your System from Malware 42

Mitigating Compromise and Abuse....45
Using Additional Application Security Practices....48
Secure Your Infrastructure....49
Using Amazon Virtual Private Cloud (VPC)....49
Using Security Zoning and Network Segmentation....51
Strengthening Network Security....54
Securing Periphery Systems: User Repositories, DNS, NTP....55
Building Threat Protection Layers....57
Test Security....60
Managing Metrics and Improvement....61
Mitigating and Protecting Against DoS & DDoS Attacks....62
Manage Security Monitoring, Alerting, Audit Trail, and Incident Response....65
Using Change Management Logs....68
Managing Logs for Critical Transactions....68
Protecting Log Information....69
Logging Faults....70
Conclusion....70
Contributors....70
Further Reading....70
Document Revisions....71

Abstract

This whitepaper is intended for existing and potential customers who are designing the security infrastructure and configuration for applications running in Amazon Web Services (AWS). It provides security best practices that will help you define your Information Security Management System (ISMS) and build a set of security policies and processes for your organization so you can protect your data and assets in the AWS Cloud. The whitepaper also provides an overview of different security topics such as identifying, categorizing and protecting your assets on AWS, managing access to AWS resources using accounts, users and groups and suggesting ways you can secure your data, your operating systems and applications and overall infrastructure in the cloud.

The paper is targeted at IT decision makers and security personnel and assumes that you are familiar with basic security concepts in the area of networking, operating systems, data encryption, and operational controls.

Introduction

Information security is of paramount importance to Amazon Web Services (AWS) customers. Security is a core functional requirement that protects mission- critical information from accidental or deliberate theft, leakage, integrity compromise, and deletion.

Under the AWS *shared responsibility model*, AWS provides a global secure infrastructure and foundation compute, storage, networking and database services, as well as higher level services. AWS provides a range of security services and features that AWS customers can use to secure their assets. AWS customers are responsible for protecting the confidentiality, integrity, and availability of their data in the cloud, and for meeting specific business requirements for information protection. For more information on AWS's security features, please read Overview of Security Processes Whitepaper.

This whitepaper describes best practices that you can leverage to build and define an Information Security Management System (ISMS), that is, a collection of information security policies and processes for your organization's assets on AWS. For more information about ISMSs, see ISO 27001 at https://www.iso.org/standard/54534.html. Although it is not required to build an ISMS to use AWS, we think that the structured approach for managing information security that is built on basic building blocks of a widely adopted global security approach will help you improve your organization's overall security posture.

We address the following topics:

- How security responsibilities are shared between AWS and you, the customer
- How to define and categorize your assets
- How to manage user access to your data using privileged accounts and groups
- Best practices for securing your data, operating systems, and network
- How monitoring and alerting can help you achieve your security objectives

This whitepaper discusses security best practices in these areas at a high level. (It does not provide "how-to" configuration guidance. For service specific configuration guidance, see the AWS Security Documentation.)

Know the AWS Shared Responsibility Model

Amazon Web Services provides a secure global infrastructure and services in the cloud. You can build your systems using AWS as the foundation, and architect an ISMS that takes advantage of AWS features.

To design an ISMS in AWS, you must first be familiar with the AWS shared responsibility model, which requires AWS and customers to work together towards security objectives.

AWS provides secure infrastructure and services, while you, the customer, are responsible for secure operating systems, platforms, and data. To ensure a secure global infrastructure, AWS configures infrastructure components and provides services and features you can use to enhance security, such as the Identity and Access Management (IAM) service, which you can use to manage users and user permissions in a subset of AWS services. To ensure secure services, AWS offers shared responsibility models for each of the different type of service that we offer:

- Infrastructure services
- Container services
- Abstracted services

The shared responsibility model for infrastructure services, such as Amazon Elastic Compute Cloud (Amazon EC2) for example, specifies that AWS manages the security of the following assets:

- Facilities
- Physical security of hardware
- Network infrastructure
- Virtualization infrastructure

Consider AWS the owner of these assets for the purposes of your ISMS asset definition. Leverage these AWS controls and include them in your ISMS.

In this Amazon EC2 example, you as the customer are responsible for the security of the following assets:

- Amazon Machine Images (AMIs)
- Operating systems

- Applications
- Data in transit
- Data at rest
- Data stores
- Credentials
- Policies and configuration

Specific services further delineate how responsibilities are shared between you and AWS. For more information, see https://aws.amazon.com/compliance/shared-responsibility-model/.

Understanding the AWS Secure Global Infrastructure

The AWS secure global infrastructure and services are managed by AWS and provide a trustworthy foundation for enterprise systems and individual applications. AWS establishes high standards for information security within the cloud, and has a comprehensive and holistic set of control objectives, ranging from physical security through software acquisition and development to employee lifecycle management and security organization. The AWS secure global infrastructure and services are subject to regular third-party compliance audits. See the Amazon Web Services Risk and Compliance whitepaper for more information.

Using the IAM Service

The IAM service is one component of the AWS secure global infrastructure that we discuss in this paper. With IAM, you can centrally manage users, security credentials such as passwords, access keys, and permissions policies that control which AWS services and resources users can access.

When you sign up for AWS, you create an AWS account, for which you have a user name (your email address) and a password. The user name and password let you log into the AWS Management Console, where you can use a browser- based interface to manage AWS resources. You can also create access keys (which consist of an access key ID and secret access key) to use when you make programmatic calls to AWS using the command line interface (CLI), the AWS SDKs, or API calls.

IAM lets you create individual users within your AWS account and give them each their own user name, password, and access keys. Individual users can then log into the

console using a URL that's specific to your account. You can also create access keys for individual users so that they can make programmatic calls to access AWS resources. All charges for activities performed by your IAM users are billed to your AWS account. As a best practice, we recommend that you create an IAM user even for yourself and that you do not use your AWS account credentials for everyday access to AWS. See Security Best Practices in IAM for more information.

Regions, Availability Zones, and Endpoints

You should also be familiar with regions, Availability Zones, and endpoints, which are components of the AWS secure global infrastructure.

Use AWS regions to manage network latency and regulatory compliance. When you store data in a specific region, it is not replicated outside that region. It is your responsibility to replicate data across regions, if your business needs require that. AWS provides information about the country, and, where applicable, the state where each region resides; you are responsible for selecting the region to store data with your compliance and network latency requirements in mind.

Regions are designed with availability in mind and consist of at least two, often more, Availability Zones. Availability Zones are designed for fault isolation. They are connected to multiple Internet Service Providers (ISPs) and different power grids. They are interconnected using high speed links, so applications can rely on Local Area Network (LAN) connectivity for communication between Availability Zones within the same region. You are responsible for carefully selecting the Availability Zones where your systems will reside. Systems can span multiple Availability Zones, and we recommend that you design your systems to survive temporary or prolonged failure of an Availability Zone in the case of a disaster.

AWS provides web access to services through the AWS Management Console, available at and then through individual consoles for each service. AWS provides programmatic access to services through Application Programming Interfaces (APIs) and command line interfaces (CLIs). Service endpoints, which are managed by AWS, provide management ("backplane") access.

Sharing Security Responsibility for AWS Services

AWS offers a variety of different infrastructure and platform services. For the purpose of understanding security and shared responsibility of these AWS services, let's categorize them in three main categories: infrastructure, container, and abstracted services. Each

category comes with a slightly different security ownership model based on how you interact and access the functionality

- **Infrastructure Services:** This category includes compute services, such as Amazon EC2, and related services, such as Amazon Elastic Block Store (Amazon EBS), Auto Scaling, and Amazon Virtual Private Cloud (Amazon VPC). With these services, you can architect and build a cloud infrastructure using technologies similar to and largely compatible with on-premises solutions. You control the operating system, and you configure and operate any identity management system that provides access to the user layer of the virtualization stack.
- **Container Services:** Services in this category typically run on separate Amazon EC2 or other infrastructure instances, but sometimes you don't manage the operating system or the platform layer. AWS provides a managed service for these application "containers". You are responsible for setting up and managing network controls, such as firewall rules, and for managing platform-level identity and access management separately from IAM. Examples of container services include Amazon Relational Database Services (Amazon RDS), Amazon Elastic Map Reduce (Amazon EMR) and AWS Elastic Beanstalk
- **Abstracted Services:** This category includes high-level storage, database, and messaging services, such as Amazon Simple Storage Service (Amazon S3), Amazon Glacier, Amazon DynamoDB, Amazon Simple Queuing Service (Amazon SQS), and Amazon Simple Email Service (Amazon SES). These services abstract the platform or management layer on which you can build and operate cloud applications. You access the endpoints of these abstracted services using AWS APIs, and AWS manages the underlying service components or the operating system on which they reside. You share the underlying infrastructure, and abstracted services provide a multi- tenant platform which isolates your data in a secure fashion and provides for powerful integration with IAM.

Let's dig a little deeper into the shared responsibility model for each service type.

Shared Responsibility Model for Infrastructure Services

Infrastructure services, such as Amazon EC2, Amazon EBS, and Amazon VPC, run on top of the AWS global infrastructure. They vary in terms of availability and durability objectives but always operate within the specific region where they have been launched. You can build systems that meet availability objectives exceeding those of

individual services from AWS by employing resilient components in multiple Availability Zones.

Figure 1 depicts the building blocks for the shared responsibility model for infrastructure services.

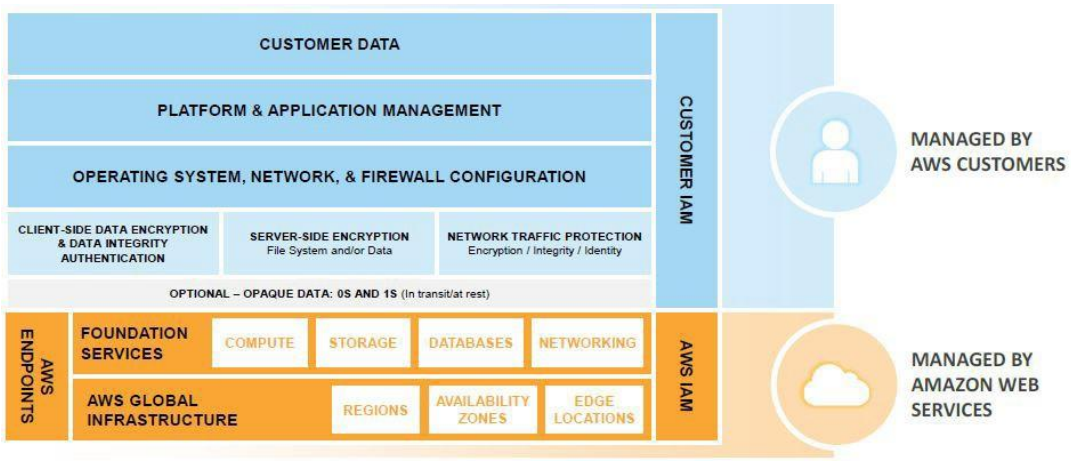

Figure 1: Shared Responsibility Model for Infrastructure Services

Building on the AWS secure global infrastructure, you install and configure your operating systems and platforms in the AWS cloud just as you would do on premises in your own data centers. Then you install your applications on your platform. Ultimately, your data resides in and is managed by your own applications. Unless you have more stringent business or compliance requirements, you don't need to introduce additional layers of protection beyond those provided by the AWS secure global infrastructure.

For certain compliance requirements, you might require an additional layer of protection between the services from AWS and your operating systems and platforms, where your applications and data reside. You can impose additional controls, such as protection of data at rest, and protection of data in transit, or introduce a layer of opacity between services from AWS and your platform. The opacity layer can include data encryption, data integrity authentication, software- and data-signing, secure time-stamping, and more.

AWS provides technologies you can implement to protect data at rest and in transit. See the Managing OS-level Access to Amazon EC2 Instances and Secure Your Data sections in this whitepaper for more information. Alternatively, you might introduce your own data protection tools, or leverage AWS partner offerings.

The previous section describes the ways in which you can manage access to resources that require authentication to AWS services. However, in order to access the operating

system on your EC2 instances, you need a different set of credentials. In the shared responsibility model, you own the operating system credentials but AWS helps you bootstrap the initial access to the operating system.

When you launch a new Amazon EC2 instance from a standard AMI, you can access that instance using secure remote system access protocols, such as Secure Shell (SSH), or Windows Remote Desktop Protocol (RDP). You must successfully authenticate at the operating-system level before you can access and configure the Amazon EC2 instance to your requirements. After you have authenticated and have remote access into the Amazon EC2 instance, you can set up the operating system authentication mechanisms you want, which might include X.509 certificate authentication, Microsoft Active Directory, or local operating system accounts.

To enable authentication to the EC2 instance, AWS provides asymmetric key pairs, known as Amazon EC2 key pairs. These are industry-standard RSA key pairs. Each user can have multiple Amazon EC2 key pairs, and can launch new instances using different key pairs. EC2 key pairs are not related to the AWS account or IAM user credentials discussed previously. Those credentials control access to other AWS services; EC2 key pairs control access only to your specific instance.

You can choose to generate your own Amazon EC2 key pairs using industry- standard tools like OpenSSL. You generate the key pair in a secure and trusted environment, and only the public key of the key pair is imported in AWS; you store the private key securely. We advise using a high-quality random number generator if you take this path.

You can choose to have Amazon EC2 key pairs generated by AWS. In this case, both the private and public key of the RSA key pair are presented to you when you first create the instance. You must download and securely store the private key of the Amazon EC2 key pair. AWS does not store the private key; if it is lost you must generate a new key pair.

For Amazon EC2 Linux instances using the **cloud-init** service, when a new instance from a standard AWS AMI is launched, the public key of the Amazon EC2 key pair is appended to the initial operating system user's

~/.ssh/authorized_keys file. That user can then use an SSH client to connect to the Amazon EC2 Linux instance by configuring the client to use the correct Amazon EC2 instance user's name as its identity (for example, ec2-user), and providing the private key file for user authentication.

For Amazon EC2 Windows instances using the **ec2config** service, when a new instance from a standard AWS AMI is launched, the **ec2config** service sets a new random Administrator password for the instance and encrypts it using the corresponding Amazon EC2 key pair's public key. The user can get the Windows instance password by using the AWS Management Console or command line tools, and by providing the corresponding Amazon EC2 private key to decrypt the password. This password, along with the default Administrative account for the Amazon EC2 instance, can be used to authenticate to the Windows instance.

AWS provides a set of flexible and practical tools for managing Amazon EC2 keys and providing industry-standard authentication into newly launched Amazon EC2 instances. If you have higher security requirements, you can implement alternative authentication mechanisms, including LDAP or Active Directory authentication, and disable Amazon EC2 key pair authentication.

Shared Responsibility Model for Container Services

The AWS shared responsibility model also applies to container services, such as Amazon RDS and Amazon EMR. For these services, AWS manages the underlying infrastructure and foundation services, the operating system and the application platform. For example, Amazon RDS for Oracle is a managed database service in which AWS manages all the layers of the container, up to and including the Oracle database platform. For services such as Amazon RDS, the AWS platform provides data backup and recovery tools; but it is your responsibility to configure and use tools in relation to your business continuity and disaster recovery (BC/DR) policy.

For AWS Container services, you are responsible for the data and for firewall rules for access to the container service. For example, Amazon RDS provides RDS security groups, and Amazon EMR allows you to manage firewall rules through Amazon EC2 security groups for Amazon EMR instances.

Figure 2 depicts the shared responsibility model for container services.

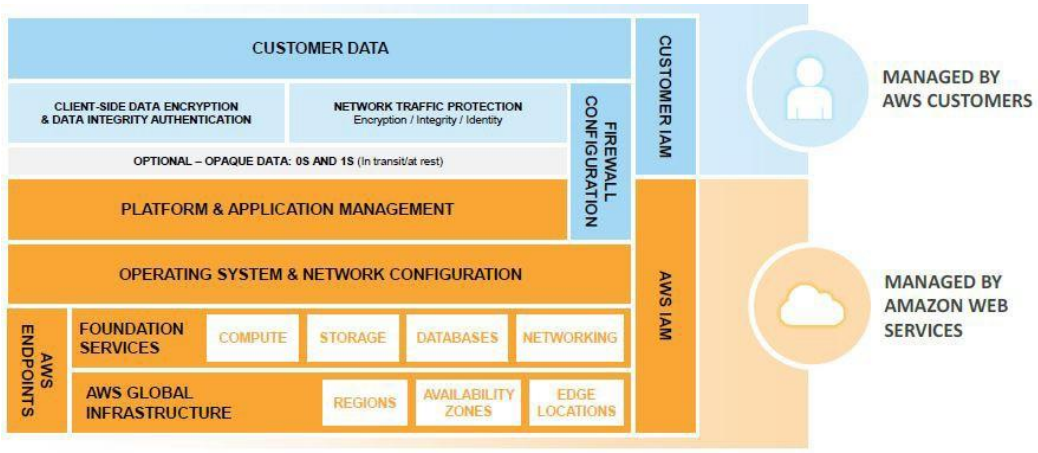

Figure 2: Shared Responsibility Model for Container Services

Shared Responsibility Model for Abstracted Services

For abstracted services, such as Amazon S3 and Amazon DynamoDB, AWS operates the infrastructure layer, the operating system, and platforms and you access the endpoints to store and retrieve data. Amazon S3 and DynamoDB are tightly integrated with IAM. You are responsible for managing your data (including classifying your assets), and for using IAM tools to apply ACL-type permissions to individual resources at the platform level, or permissions based on user identity or user responsibility at the IAM user/group level. For some services, such as Amazon S3, you can also use platform-provided encryption of data at rest, or platform-provided HTTPS encapsulation for your payloads for protecting your data in transit to and from the service.

Figure 3 outlines the shared responsibility model for AWS abstracted services:

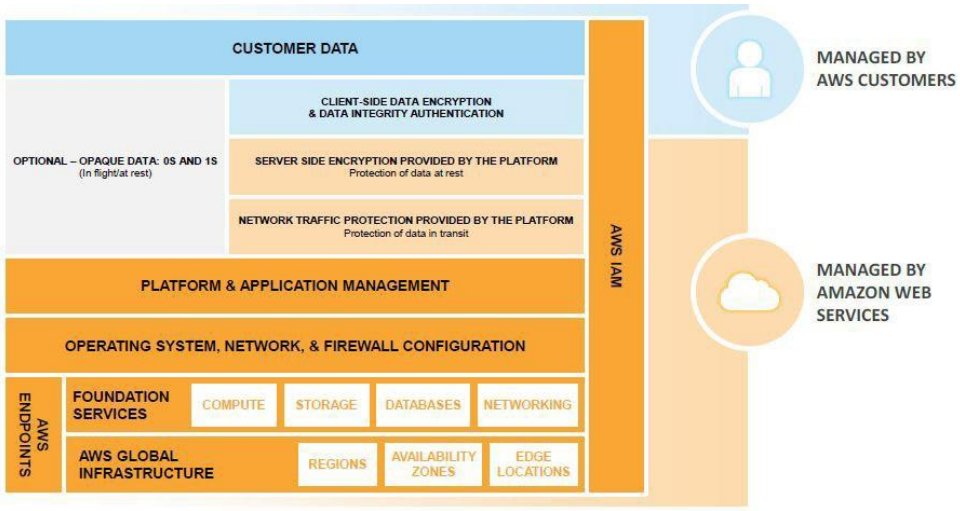

Figure 3: Shared Responsibility Model for Abstracted Services

Using the Trusted Advisor Tool

Some AWS Premium Support plans include access to the Trusted Advisor tool, which offers a one-view snapshot of your service and helps identify common security misconfigurations, suggestions for improving system performance, and underutilized resources. In this whitepaper we cover the security aspects of Trusted Advisor that apply to Amazon EC2.

Trusted Advisor checks for compliance with the following security recommendations:

- Limited access to common administrative ports to only a small subset of addresses. This includes ports 22 (SSH), 23 (Telnet) 3389 (RDP), and 5500 (VNC).
- Limited access to common database ports. This includes ports 1433 (MSSQL Server), 1434 (MSSQL Monitor), 3306 (MySQL), Oracle (1521) and 5432 (PostgreSQL).
- IAM is configured to help ensure secure access control of AWS resources.
- Multi-factor authentication (MFA) token is enabled to provide two-factor authentication for the root AWS account.

Define and Categorize Assets on AWS

Before you design your ISMS, identify all the information assets that you need to protect and then devise a technically and financially viable solution for protecting them. It can be difficult to quantify every asset in financial terms, so you might find that using qualitative metrics (such as negligible/low/medium/high/very high) is a better option.

Assets fall into two categories:

- Essential elements, such as business information, process, and activities
- Components that support the essential elements, such as hardware, software, personnel, sites, and partner organizations

Table 1 shows a sample matrix of assets.

Table 1: Sample asset matrix

Asset Name	Asset Owner	Asset Category	Dependencies
Customer-facing website applications	E-Commerce team	Essential	EC2, Elastic Load Balancing, Amazon RDS, development
Customer credit card data	E-Commerce team	Essential	PCI card holder environment, encryption, AWS PCI service
Personnel data	COO	Essential	Amazon RDS, encryption provider, dev and ops IT, third party
Data archive	COO	Essential	S3, S3 Glacier, dev and ops IT
HR management system	HR	Essential	EC2, S3, RDS, dev and ops IT, third party
AWS Direct Connect infrastructure	CIO	Network	Network ops, TelCo provider, AWS Direct Connect
Business intelligence platform	BI team	Software	EMR, Redshift, DynamoDB, S3, dev and ops
Business intelligence services	COO	Essential	BI infrastructure, BI analysis teams
LDAP directory	IT Security team	Security	EC2, IAM, custom software, dev and ops
Windows AMI	Server team	Software	EC2, patch management software, dev and ops
Customer credentials	Compliance team	Security	Daily updates; archival infrastructure

Design Your ISMS to Protect Your Assets on AWS

After you have determined assets, categories, and costs, establish a standard for implementing, operating, monitoring, reviewing, maintaining, and improving your information security management system (ISMS) on AWS. Security requirements differ in every organization, depending on the following factors:

- Business needs and objectives
- Processes employed
- Size and structure of the organization

All these factors can change over time, so it is a good practice to build a cyclical process for managing all of this information. Table 2 suggests a phased approach to designing and building an ISMS in AWS. You might also find standard frameworks, such as ISO 27001, helpful with ISMS design and implementation.

Table 2: Phases of building an ISMS

Phase	Title	Description
1	Define scope and boundaries.	Define which regions, Availability Zones, instances and AWS resources are "in scope." If you exclude any component (for example, AWS manages facilities, so you can leave it out of your own management system), state what you have excluded and why explicitly.
2	Define an ISMS policy.	Include the following: • Objectives that set the direction and principles for action regarding information security • Legal, contractual, and regulatory requirements • Risk management objectives for your organization • How you will measure risk • How management approves the plan
3	Select a risk assessment methodology.	Select a risk assessment methodology based on input from groups in your organization about the following factors: • Business needs • Information security requirements • Information technology capabilities and use • Legal requirements • Regulatory responsibilities Because public cloud infrastructure operates differently from legacy environments, it is critical to set criteria for accepting risks and identifying the acceptable levels of risk (risk tolerances). We recommended starting with a risk assessment and leveraging automation as much as possible. AWS risk automation can narrow down the scope of resources required for risk management. There are several risk assessment methodologies, including OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation), ISO 31000:2009 Risk Management, ENISA (European Network and Information Security Agency, IRAM (Information Risk Analysis Methodology), and NIST (National Institute of Standards & Technology) Special Publication (SP) 800-30 rev.1 Risk Management Guide.

Phase	Title	Description
4	Identify risks	We recommend that you create a risk register by mapping all your assets to threats, and then, based on the vulnerability assessment and impact analysis results, creating a new risk matrix for each AWS environment. Here's an example risk register: • Assets • Threats to those assets • Vulnerabilities that could be exploited by those threats • Consequences if those vulnerabilities are exploited
5	Analyze and evaluate risks	Analyze and evaluate the risk by calculating business impact, likelihood and probability, and risk levels.
6	Address risks	Select options for addressing risks. Options include applying security controls, accepting risks, avoiding risk, or transferring risks.
7	Choose a security control framework	When you choose your security controls, use a framework, such as ISO 27002, NIST SP 800-53, COBIT (Control Objectives for Information and related Technology) and CSA-CCM (Cloud Security Alliance-Cloud Control Matrix. These frameworks comprise a set of reusable best practices and will help you to choose relevant controls.
8	Get management approval	Even after you have implemented all controls, there will be residual risk. We recommend that you get approval from your business management that acknowledges all residual risks, and approvals for implementing and operating the ISMS.
9	Statement of applicability	Create a statement of applicability that includes the following information: • Which controls you chose and why • Which controls are in place • Which controls you plan to put in place • Which controls you excluded and why

Manage AWS Accounts, IAM Users, Groups, and Roles

Ensuring that users have appropriate levels of permissions to access the resources they need, but no more than that, is an important part of every ISMS. You can use IAM to help perform this function. You *create IAM users* under your AWS account and then assign them permissions directly, or assign them to groups to which you assign permissions. Here's a little more detail about AWS accounts and IAM users:

- **AWS account.** This is the account that you create when you first sign up for AWS. Your AWS account represents a business relationship between you and AWS. You use your AWS account to manage your AWS resources and services. AWS accounts have root permissions to all AWS resources and services, so they are very powerful. Do not use root account credentials for day-to-day interactions with AWS. In some cases, your organization might choose to use several AWS accounts, one for each major department, for example, and then create IAM users within each of the AWS accounts for the appropriate people and resources.
- **IAM users.** With IAM you can create multiple users, each with individual security credentials, all controlled under a single AWS account. IAM users can be a person, service, or application that needs access to your AWS resources through the management console, CLI, or directly via APIs. Best practice is to create individual IAM users for each individual that needs to access services and resources in your AWS account. You can create fine- grained permissions to resources under your AWS account, apply them to groups you create, and then assign users to those groups. This best practice helps ensure users have least privilege to accomplish tasks.

Strategies for Using Multiple AWS Accounts

Design your AWS account strategy to maximize security and follow your business and governance requirements. Table 3 discusses possible strategies.

Table 3: AWS Account strategies

Business Requirement	Proposed Design	Comments
Centralized security management	Single AWS account	Centralize information security management and minimize overhead.
Separation of production, development, and testing environments	Three AWS accounts	Create one AWS account for production services, one for development, and one for testing.
Multiple autonomous departments	Multiple AWS accounts	Create separate AWS accounts for each autonomous part of the organization. You can assign permissions and policies under each account.

Business Requirement	Proposed Design	Comments
Centralized security management with multiple autonomous independent projects	Multiple AWS accounts	Create a single AWS account for common project resources (such as DNS services, Active Directory, CMS etc.).Then create separate AWS accounts per project. You can assign permissions and policies under each project account and grant access to resources across accounts.

You can configure a consolidated billing relationship across multiple accounts to ease the complexity of managing a different bill for each account and leverage economies of scale. When you use billing consolidation, the resources and credentials are not shared between accounts.

Managing IAM Users

IAM users with the appropriate level of permissions can create new IAM users, or manage and delete existing ones. This highly privileged IAM user can create a distinct IAM user for each individual, service, or application within your organization that manages AWS configuration or accesses AWS resources directly. We strongly discourage the use of shared user identities, where multiple entities share the same credentials.

Managing IAM Groups

IAM groups are collections of IAM users in one AWS account. You can create IAM groups on a functional, organizational, or geographic basis, or by project, or on any other basis where IAM users need to access similar AWS resources to do their jobs. You can provide each IAM group with permissions to access AWS resources by assigning one or more IAM policies. All policies assigned to an IAM group are inherited by the IAM users who are members of the group.

For example, let’s assume that IAM user John is responsible for backups within an organization, and needs to access objects in the Amazon S3 bucket called Archives. You can give John permissions directly so he can access the Archives bucket. But then your organization places Sally and Betty on the same team as John. While you can assign user permissions individually to John, Sally, and Betty to give them access to the Archives bucket, assigning the permissions to a group and placing John, Sally, and Betty in that group will be easier to manage and maintain. If additional users require the same access, you can give it to them by adding them to the group. When a user no

longer needs access to a resource, you can remove them from the groups that provide access to that resource.

IAM groups are a powerful tool for managing access to AWS resources. Even if you only have one user who requires access to a specific resource, as a best practice, you should identify or create a new AWS group for that access, and provision user access via group membership, as well as permissions and policies assigned at the group level.

Managing AWS Credentials

Each AWS account or IAM user is a unique identity and has unique long-term credentials. There are two primary types of credentials associated with these identities: (1) those used for sign-in to the AWS Management Console and AWS portal pages, and (2) those used for programmatic access to the AWS APIs. Table 4 describes two types of sign-in credentials.

Table 4: Sign-in credentials

Sign-In Credential Type	Details
Username/Password	User names for AWS accounts are always email addresses. IAM user names allow for more flexibility. Your AWS account password can be anything you define. IAM user passwords can be forced to comply with a policy you define (that is, you can require minimum password length or the use of non-alphanumeric characters).
Multi-factor authentication (MFA)	AWS Multi-factor authentication (MFA) provides an extra level of security for sign-in credentials. With MFA enabled, when users signs in to an AWS website, they will be prompted for their user name and password (the first factor–what they know), as well as for an authentication code from their MFA device (the second factor– what they have). You can also require MFA for users to delete S3 objects. We recommend you activate MFA for your AWS account and your IAM users to prevent unauthorized access to your AWS environment. Currently AWS supports Gemalto hardware MFA devices as well as virtual MFA devices in the form of smartphone applications.

Table 5 describes types of credentials used for programmatic access to APIs.

Table 5: API access credentials

Access Credential Type	Details
Access keys	Access keys are used to digitally sign API calls made to AWS services. Each access key credential is comprised of an access key ID and a secret key. The secret key portion must be secured by the AWS account holder or the IAM user to whom they are assigned. Users can have two sets of active access keys at any one time. As a best practice, users should rotate their access keys on a regular basis.
MFA for API calls	Multi-factor authentication (MFA)-protected API access requires IAM users to enter a valid MFA code before they can use certain functions, which are APIs. Policies you create in IAM will determine which APIs require MFA. Because the AWS Management Console calls AWS service APIs, you can enforce MFA on APIs whether access is through the console or via APIs.

Understanding Delegation Using IAM Roles and Temporary Security Credentials

There are scenarios in which you want to delegate access to users or services that don't normally have access to your AWS resources. Table 6 below outlines common use cases for delegating such access.

Table 6: Common delegation use cases

Use Case	Description
Applications running on Amazon EC2 instances that need to access AWS resources	Applications that run on an Amazon EC2 instance and that need access to AWS resources such as Amazon S3 buckets or an Amazon DynamoDB table must have security credentials in order to make programmatic requests to AWS. Developers might distribute their credentials to each instance and applications can then use those credentials to access resources, but distributing long-term credentials to each instance is challenging to manage and a potential security risk.

Use Case	Description
Cross account access	To manage access to resources, you might have multiple AWS accounts—for example, to isolate a development environment from a production environment. However, users from one account might need to access resources in the other account, such as promoting an update from the development environment to the production environment. Although users who work in both accounts could have a separate identity in each account, managing credentials for multiple accounts makes identity management difficult.
Identity federation	Users might already have identities outside of AWS, such as in your corporate directory. However, those users might need to work with AWS resources (or work with applications that access those resources). If so, these users also need AWS security credentials in order to make requests to AWS.

IAM roles and temporary security credentials address these use cases. An IAM role lets you define a set of permissions to access the resources that a user or service needs, but the permissions are not attached to a specific IAM user or group. Instead, IAM users, mobile and EC2-based applications, or AWS services (like Amazon EC2) can programmatically assume a role. Assuming the role returns temporary security credentials that the user or application can use to make for programmatic requests to AWS. These temporary security credentials have a configurable expiration and are automatically rotated. Using IAM roles and temporary security credentials means you don't always have to manage long- term credentials and IAM users for each entity that requires access to a resource.

IAM Roles for Amazon EC2

IAM Roles for Amazon EC2 is a specific implementation of IAM roles that addresses the first use case in Table 6. In the following figure, a developer is running an application on an Amazon EC2 instance that requires access to the Amazon S3 bucket named photos. An administrator creates the Get-pics role. The role includes policies that grant read permissions for the bucket and that allow the developer to launch the role with an Amazon EC2 instance. When the application runs on the instance, it can access the photos bucket by using the role's temporary credentials. The administrator doesn't have to grant the developer permission to access the photos bucket, and the developer never has to share credentials.

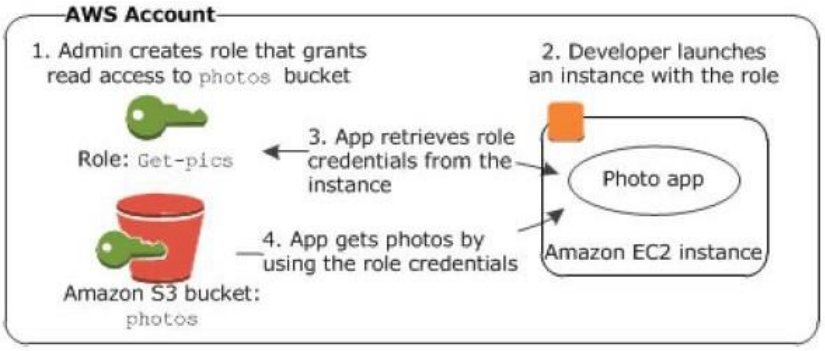

Figure 4: How roles for EC2 work

1. An administrator uses IAM to create the `Get-pics` role. In the role, the administrator uses a policy that specifies that only Amazon EC2 instances can assume the role and that specifies only read permissions for the photos bucket.
2. A developer launches an Amazon EC2 instance and associates the `Get-pics` role with that instance.
3. When the application runs, it retrieves credentials from the instance metadata on the Amazon EC2 instance.
4. Using the role credentials, the application accesses the `photo` bucket with read-only permissions.

Cross-Account Access

You can use IAM roles to address the second use case in Table 6 by enabling IAM users from another AWS account to access resources within your AWS account. This process is referred to as cross-account access. Cross-account access lets you share access to your resources with users in other AWS accounts.

To establish cross-account access, in the trusting account (Account A), you create an IAM policy that grants the trusted account (Account B) access to specific resources. Account B can then delegate this access to its IAM users. Account B cannot delegate more access to its IAM users than the permissions that it has been granted by Account A.

Identity Federation

You can use IAM roles to address the third use case in Table 6 by creating an identity broker that sits between your corporate users and your AWS resources to manage the authentication and authorization process without needing to re- create all your users as IAM users in AWS.

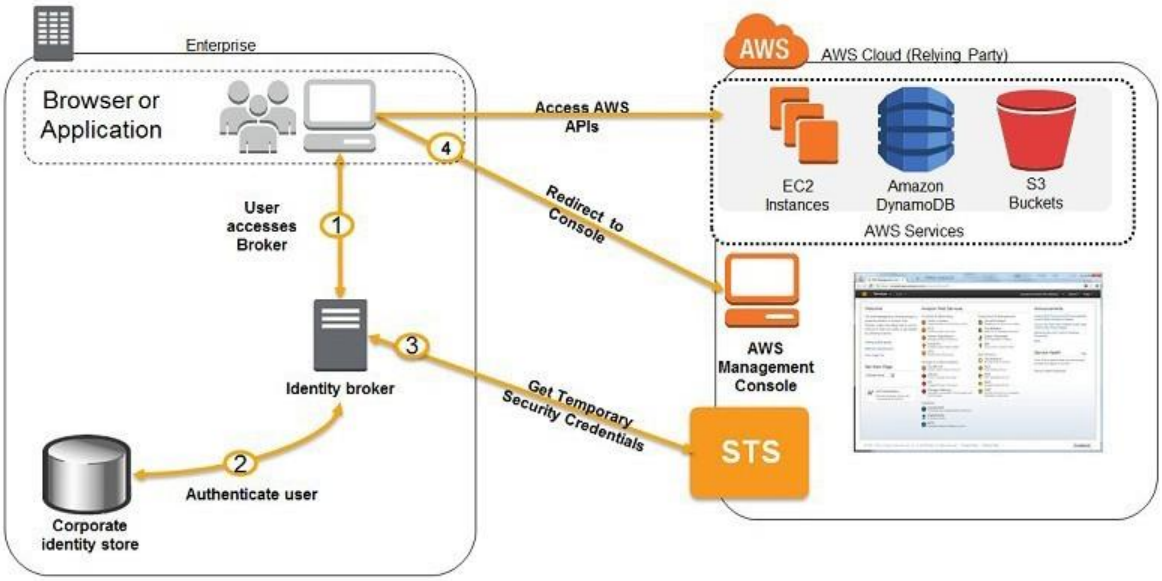

Figure 5: AWS identity federation with temporary security credentials

1. The enterprise user accesses the identity broker application.
2. The identity broker application authenticates the users against the corporate identity store.
3. The identity broker application has permissions to access the AWS Security Token Service (STS) to request temporary security credentials.
4. Enterprise users can get a temporary URL that gives them access to the AWS APIs or the Management Console. A sample identity broker application for use with Microsoft Active Directory is provided by AWS.

Managing OS-level Access to Amazon EC2 Instances

The previous section describes the ways in which you can manage access to resources that require authentication to AWS services. However, in order to access the operating system on your EC2 instances, you need a different set of credentials. In the shared responsibility model, you own the operating system credentials but AWS helps you bootstrap the initial access to the operating system.

When you launch a new Amazon EC2 instance from a standard AMI, you can access that instance using secure remote system access protocols, such as Secure Shell (SSH), or Windows Remote Desktop Protocol (RDP). You must successfully

authenticate at the operating-system level before you can access and configure the Amazon EC2 instance to your requirements. After you have authenticated and have remote access into the Amazon EC2 instance, you can set up the operating system authentication mechanisms you want, which might include X.509 certificate authentication, Microsoft Active Directory, or local operating system accounts.

To enable authentication to the EC2 instance, AWS provides asymmetric key pairs, known as Amazon EC2 key pairs. These are industry-standard RSA key pairs. Each user can have multiple Amazon EC2 key pairs, and can launch new instances using different key pairs. EC2 key pairs are not related to the AWS account or IAM user credentials discussed previously. Those credentials control access to other AWS services; EC2 key pairs control access only to your specific instance.

You can choose to generate your own Amazon EC2 key pairs using industry- standard tools like OpenSSL. You generate the key pair in a secure and trusted environment, and only the public key of the key pair is imported in AWS; you store the private key securely. We advise using a high-quality random number generator if you take this path.

You can choose to have Amazon EC2 key pairs generated by AWS. In this case, both the private and public key of the RSA key pair are presented to you when you first create the instance. You must download and securely store the private key of the Amazon EC2 key pair. AWS does not store the private key; if it is lost you must generate a new key pair.

For Amazon EC2 Linux instances using the **cloud-init** service, when a new instance from a standard AWS AMI is launched, the public key of the Amazon EC2 key pair is appended to the initial operating system user's **~/.ssh/authorized_keys** file. That user can then use an SSH client to connect to the Amazon EC2 Linux instance by configuring the client to use the correct Amazon EC2 instance user's name as its identity (for example, ec2-user), and providing the private key file for user authentication.

For Amazon EC2 Windows instances using the **ec2config** service, when a new instance from a standard AWS AMI is launched, the **ec2config** service sets a new random Administrator password for the instance and encrypts it using the corresponding Amazon EC2 key pair's public key. The user can get the Windows instance password by using the AWS Management Console or command line tools, and by providing the corresponding Amazon EC2 private key to decrypt the password. This password, along with the default Administrative account for the Amazon EC2 instance, can be used to authenticate to the Windows instance.

AWS provides a set of flexible and practical tools for managing Amazon EC2 keys and providing industry-standard authentication into newly launched Amazon EC2 instances. If you have higher security requirements, you can implement alternative authentication mechanisms, including LDAP or Active Directory authentication, and disable Amazon EC2 key pair authentication.

Secure Your Data

This section discusses protecting data at rest and in transit on the AWS platform. We assume that you have already identified and classified your assets and established protection objectives for them based on their risk profiles.

Resource Access Authorization

After a user or IAM role has been authenticated, they can access resources to which they are authorized. You provide resource authorization using resource policies or capability policies, depending on whether you want the user to have control over the resources, or whether you want to override individual user control.

- **Resource policies** are appropriate in cases where the user creates resources and then wants to allow other users to access those resources. In this model the policy is attached directly to the resource and describes who can do what with the resource. The user is in control of the resource. You can provide an IAM user with explicit access to a resource. The root AWS account always has access to manage resource policies, and is the owner of all resources created in that account. Alternatively, you can grant users explicit access to manage permissions on a resource.
- **Capability policies** (which in the IAM docs are referred to as "user- based permissions") are often used to enforce company-wide access policies. Capability policies are assigned to an IAM user either directly or indirectly using an IAM group. They can also be assigned to a role that will be assumed at run time. Capability policies define what capabilities (actions) the user is allowed or denied to perform. They can override resource-based policies permissions by explicitly denying them.
- IAM policies can be used to restrict access to a specific source IP address range, or during specific days and times of the day, as well as based on other conditions.

- Resource policies and capability policies and are cumulative in nature: An individual user's effective permissions is the union of a resources policies and the capability permissions granted directly or through group membership.

Storing and Managing Encryption Keys in the Cloud

Security measures that rely on encryption require keys. In the cloud, as in an on-premises system, it is essential to keep your keys secure.

You can use existing processes to manage encryption keys in the cloud, or you can leverage server-side encryption with AWS key management and storage capabilities.

If you decide to use your own key management processes, you can use different approaches to store and protect key material. We strongly recommend that you store keys in tamper-proof storage, such as Hardware Security Modules. Amazon Web Services provides an HSM service in the cloud, known as AWS CloudHSM. Alternatively, you can use HSMs that store keys on premises, and access them over secure links, such as IPSec virtual private networks (VPNs) to Amazon VPC, or AWS Direct Connect with IPSec.

You can use on-premises HSMs or CloudHSM to support a variety of use cases and applications, such as database encryption, Digital Rights Management (DRM), and Public Key Infrastructure (PKI) including authentication and authorization, document signing, and transaction processing. CloudHSM currently uses Luna SA HSMs from SafeNet. The Luna SA is designed to meet Federal Information Processing Standard (FIPS) 140-2 and Common Criteria EAL4+ standards, and supports a variety of industry-standard cryptographic algorithms.

When you sign up for CloudHSM, you receive dedicated single tenant access to CloudHSM appliances. Each appliance appears as a resource in your Amazon VPC. You, not AWS, initialize and manage the cryptographic domain of the CloudHSM. The cryptographic domain is a logical and physical security boundary that restricts access to your keys. Only you can control your keys and operations performed by the CloudHSM. AWS administrators manage, maintain, and monitor the health of the CloudHSM appliance, but do not have access to the cryptographic domain. After you initialize the cryptographic domain, you can configure clients on your EC2 instances that allow applications to use the APIs provided by CloudHSM.

Your applications can use the standard APIs supported by the CloudHSM, such as PKCS#11, MS CAPI, and Java JCA/JCE (Java Cryptography Architecture/Java Cryptography Extensions). The CloudHSM client provides the APIs to your applications

and implements each API call by connecting to the CloudHSM appliance using a mutually authenticated SSL connection.

You can implement CloudHSMs in multiple Availability Zones with replication between them to provide for high availability and storage resilience.

Protecting Data at Rest

For regulatory or business requirement reasons, you might want to further protect your data at rest stored in Amazon S3, on Amazon EBS, Amazon RDS, or other services from AWS. Table 7 lists concern to consider when you are implementing protection of data at rest on AWS.

Table 7: Threats to data at rest

Concern	Recommended Protection Approach	Strategies
Accidental information disclosure	Designate data as confidential and limit the number of users who can access it. Use AWS permissions to manage access to resources for services such as Amazon S3. Use encryption to protect confidential data on Amazon EBS, or Amazon RDS.	Permissions File, partition, volume or application-level encryption
Data integrity compromise	To ensure that data integrity is not compromised through deliberate or accidental modification, use resource permissions to limit the scope of users who can modify the data. Even with resource permissions, accidental deletion by a privileged user is still a threat (including a potential attack by a Trojan using the privileged user's credentials), which illustrates the importance of the principle of least privilege. Perform data integrity checks, such as Message Authentication Codes (SHA-1/SHA-2), or Hashed Message Authentication Codes (HMACs), digital signatures, or authenticated encryption (AES-GCM), to detect data integrity compromise. If you detect data compromise, restore the data from backup, or, in the case of Amazon S3, from a previous object version.	Permissions Data integrity checks (MAC/HMAC/Digital Signatures/Authenticated Encryption) Backup Versioning (Amazon S3)

Concern	Recommended Protection Approach	Strategies
Accidental deletion	Using the correct permissions and the rule of the least privilege is the best protection against accidental or malicious deletion. For services such as Amazon S3, you can use MFA Delete to require multi-factor authentication to delete an object, limiting access to Amazon S3 objects to privileged users. If you detect data compromise, restore the data from backup, or, in the case of Amazon S3, from a previous object version.	Permissions Backup Versioning (Amazon S3) MFA Delete (Amazon S3)
System, infrastructure, hardware or software availability	In the case of a system failure or a natural disaster, restore your data from backup, or from replicas. Some services, such as Amazon S3 and Amazon DynamoDB, provide automatic data replication between multiple Availability Zones within a region. Other services require you to configure replication or backups.	Backup Replication

Analyze the threat landscape that applies to you, and employ the relevant protection techniques as outlined in *Table 1*.

The following sections describe how you can configure different services from AWS to protect data at rest.

Protecting Data at Rest on Amazon S3

Amazon S3 provides a number of security features for protection of data at rest, which you can use or not depending on your threat profile. Table 8 summarizes these features:

Table 8: Amazon S3 features for protecting data at rest

Amazon S3 Feature	Description
Permissions	Use bucket-level or object-level permissions alongside IAM policies to protect resources from unauthorized access and to prevent information disclosure, data integrity compromise, or deletion.
Versioning	Amazon S3 supports object versions. Versioning is disabled by default. Enable versioning to store a new version for every modified or deleted object from which you can restore compromised objects if necessary.

Amazon S3 Feature	Description
Replication	Amazon S3 replicates each object across all Availability Zones within the respective region. Replication can provide data and service availability in the case of system failure, but provides no protection against accidental deletion or data integrity compromise–it replicates changes across all Availability Zones where it stores copies. Amazon S3 offers standard redundancy and reduced redundancy options, which have different durability objectives and price points.
Backup	Amazon S3 supports data replication and versioning instead of automatic backups. You can, however, use application-level technologies to back up data stored in Amazon S3 to other AWS regions or to on-premises backup systems.
Encryption–server side	Amazon S3 supports server-side encryption of user data. Server-side encryption is transparent to the end user. AWS generates a unique encryption key for each object, and then encrypts the object using AES-256. The encryption key is then encrypted itself using AES-256-with a master key that is stored in a secure location. The master key is rotated on a regular basis.
Encryption–client side	With client-side encryption you create and manage your own encryption keys. Keys you create are not exported to AWS in clear text. Your applications encrypt data before submitting it to Amazon S3, and decrypt data after receiving it from Amazon S3. Data is stored in an encrypted form, with keys and algorithms only known to you. While you can use any encryption algorithm, and either symmetric or asymmetric keys to encrypt the data, the AWS-provided Java SDK offers Amazon S3 client-side encryption features. See Further Reading for more information.

Protecting Data at Rest on Amazon EBS

Amazon EBS is the AWS abstract block storage service. You receive each Amazon EBS volume in raw, unformatted mode, as if it were a new hard disk. You can partition the Amazon EBS volume, create software RAID arrays, format the partitions with any file system you choose, and ultimately protect the data on the Amazon EBS volume. All of these decisions and operations on the Amazon EBS volume are opaque to AWS operations.

You can attach Amazon EBS volumes to Amazon EC2 instances. Table 9 summarizes features for protecting Amazon EBS data at rest with the operating system running on an Amazon EC2 instance.

Table 9: Amazon EBS features for protecting data at rest

Amazon EBS Feature	Description
Replication	Each Amazon EBS volume is stored as a file, and AWS creates two copies of the EBS volume for redundancy. Both copies reside in the same Availability Zone, however, so while Amazon EBS replication can survive hardware failure; it is not suitable as an availability tool for prolonged outages or disaster recovery purposes. We recommend that you replicate data at the application level, and/or create backups.
Backup	Amazon EBS provides snapshots that capture the data stored on an Amazon EBS volume at a specific point in time. If the volume is corrupt (for example, due to system failure), or data from it is deleted, you can restore the volume from snapshots. Amazon EBS snapshots are AWS objects to which IAM users, groups, and roles can be assigned permissions, so that only authorized users can access Amazon EBS backups.
Encryption: Microsoft Windows EFS	If you are running Microsoft Windows Server on AWS and you require an additional level of data confidentiality, you can implement Encrypted File System (EFS) to further protect sensitive data stored on system or data partitions. EFS is an extension to the NTFS file system that provides for transparent file and folder encryption, and integrates with Windows and Active Directory key management facilities, and PKI. You can manage your own keys on EFS.
Encryption: Microsoft	Windows BitLocker is a volume (or partition, in the case of single drive) encryption solution included in Windows Server 2008 and later operating systems. BitLocker uses
Windows Bitlocker	AES 128- and 256-bit encryption. By default, BitLocker requires a Trusted Platform Module (TPM) to store keys; this is not supported on Amazon EC2. However, you can protect EBS volumes using BitLocker, if you configure it to use a password.
Encryption: Linux dm-crypt	On Linux instances running kernel versions 2.6 and later, you can use dm-crypt to configure transparent data encryption on Amazon EBS volumes and swap space. You can use various ciphers, as well as Linux Unified Key Setup (LUKS), for key management.
Encryption: TrueCrypt	TrueCrypt is a third-party tool that offers transparent encryption of data at rest on Amazon EBS volumes. TrueCrypt supports both Microsoft Windows and Linux operating systems.
Encryption and integrity authentication: SafeNet ProtectV	SafeNet ProtectV is a third-party offering that allows for full disk encryption of Amazon EBS volumes and pre-boot authentication of AMIs. SafeNet ProtectV provides data confidentiality and data integrity authentication for data and the underlying operating system.

Protecting Data at Rest on Amazon RDS

Amazon RDS leverages the same secure infrastructure as Amazon EC2. You can use the Amazon RDS service without additional protection, but if you require encryption or data integrity authentication of data at rest for compliance or other purposes, you can add protection at the application layer, or at the platform layer using SQL cryptographic functions.

You could add protection at the application layer, for example, using a built-in encryption function that encrypts all sensitive database fields, using an application key, before storing them in the database. The application can manage keys by using symmetric encryption with PKI infrastructure or other asymmetric key techniques to provide for a master encryption key.

You could add protection at the platform using MySQL cryptographic functions; which can take the form of a statement like the following:

```
INSERT INTO Customers (CustomerFirstName,CustomerLastName) VALUES
(AES_ENCRYPT('John',@key),AES_ENCRYPT('Smith',@key);
```

Platform-level encryption keys would be managed at the application level, like application-level encryption keys. Table 10 summarizes Amazon RDS platform- level protection options.

Table 10: Amazon RDS platform level data protection at rest

Amazon RDS Platform	Comment
MySQL	MySQL cryptographic functions include encryption, hashing, and compression. For more information, see https://dev.mysql.com/doc/refman/5.5/en/encryption-functions.html.
Oracle	Oracle Transparent Data Encryption is supported on Amazon RDS for Oracle Enterprise Edition under the Bring Your Own License (BYOL) model.
Microsoft SQL	Microsoft Transact-SQL data protection functions include encryption, signing, and hashing. For more information, see http://msdn.microsoft.com/en-us/library/ms173744.

Note that SQL range queries are no longer applicable to the encrypted portion of the data. This query, for example, would not return the expected results for names like “John”, “Jonathan,” and “Joan” if the contents of column `CustomerFirstName` is encrypted at the application or platform layer:

```
SELECT CustomerFirstName, CustomerLastName from Customers WHERE
CustomerName LIKE 'Jo%';”
```

Direct comparisons, such as the following, would work and return the expected result for all fields where `CustomerFirstName` matches “John” exactly.

```
SELECT CustomerFirstName, CustomerLastName FROM Customers WHERE
CustomerFirstName = AES_ENCRYPT('John', @key);
```

Range queries would also work on fields that are not encrypted. For example, a Date field in a table could be left unencrypted so that you could use it in range queries.

One-way functions are a good way to obfuscate personal identifiers, such as social security numbers or equivalent personal IDs where they are used as unique identifiers. While you can encrypt personal identifiers, and decrypt them at the application or platform layer before using them, it’s more convenient to use a one-way function, such as keyed HMAC-SHA1, to convert the personal identifier to a fixed-length hash value. The personal identifier is still unique, because collisions in commercial HMACs are extremely rare. The HMAC is not reversible to the original personal identifier, however, so you cannot track back the data to the original individual unless you know the original personal ID, and process it via the same keyed HMAC function.

In all regions, Amazon RDS supports Transparent Data Encryption and Native Network Encryption, both of which are components of the Advanced Security option for the Oracle Database 11g Enterprise Edition. Oracle Database 11g Enterprise Edition is available on Amazon RDS for Oracle under the Bring-Your- Own-License (BYOL) model. There is no additional charge to use these features.

Oracle Transparent Data Encryption encrypts data before it is written to storage and decrypts data when it is read from storage. With Oracle Transparent Data Encryption you can encrypt table spaces or specific table columns using industry- standard encryption algorithms such as Advanced Encryption Standard (AES) and Data Encryption Standard (Triple DES).

Protecting Data at Rest on Amazon S3 Glacier

Data at rest stored in Amazon S3 Glacier is automatically server-side encrypted using 256-bit Advanced Encryption Standard (AES-256) with keys maintained by AWS. The encryption key is then encrypted itself using AES-256 with a master key that is stored in

a secure location. The master key is rotated on a regular basis. For more information about the default encryption behavior for an Amazon S3 bucket, see Amazon S3 Default Encryption.

Protecting Data at Rest on Amazon DynamoDB

Amazon DynamoDB is a shared service from AWS. You can use DynamoDB without adding protection, but you can also implement a data encryption layer over the standard DynamoDB service. See the previous section for considerations for protecting data at the application layer, including impact on range queries.

DynamoDB supports number, string, and raw binary data type formats. When storing encrypted fields in DynamoDB, it is a best practice to use raw binary fields or Base64-encoded string fields.

Protecting Data at Rest on Amazon EMR

Amazon EMR is a managed service in the cloud. AWS provides the AMIs required to run Amazon EMR, and you can't use custom AMIs or your own EBS volumes. By default, Amazon EMR instances do not encrypt data at rest.

Amazon EMR clusters often use either Amazon S3 or DynamoDB as the persistent data store. When an Amazon EMR cluster starts, it can copy the data required for it to operate from the persistent store into HDFS, or use data directly from Amazon S3 or DynamoDB.

To provide for a higher level of data at rest confidentiality or integrity you can employ a number of techniques, summarized in Table 11.

Table 11: Protecting data at rest in Amazon EMR

Requirement	Description
Amazon S3 server-side encryption–no HDFS copy	Data is permanently stored on Amazon S3 only, and not copied to HDFS at all. Hadoop fetches data from Amazon S3 and processes it locally without making persistent local copies. See the Protecting Data at Rest on Amazon S3 section for more information on Amazon S3 server-side encryption.

Requirement	Description
Amazon S3 client-side encryption	Data is permanently stored on Amazon S3 only, and not copied to HDFS at all. Hadoop fetches data from Amazon S3 and processes it locally without making persistent local copies. To apply client-side decryption, you can use a custom Serializer/Deserializer (SerDe) with products such as Hive, or InputFormat for Java Map Reduce jobs. Apply encryption at each individual row or record, so that you can split the file. See the Protecting Data at Rest on Amazon S3 section for more information on Amazon S3 client-side encryption.
Application-level encryption–entire file encrypted	You can encrypt, or protect the integrity of the data (for example, by using HMAC-SHA1) at the application level while you store data in Amazon S3 or DynamoDB. To decrypt the data, you would use a custom SerDe with Hive, or a script or a bootstrap action to fetch the data from Amazon S3, decrypt it, and load it into HDFS before processing. Because the entire file is encrypted, you might need to execute this action on a single node, such as the master node. You can use tools such as S3Distcp with special codecs.
Application-level encryption–individual fields encrypted/structure preserved	Hadoop can use a standard SerDe, such as JSON. Data decryption can take place during the Map stage of the Hadoop job, and you can use standard input/output redirection via custom decryption tools for streaming jobs.
Hybrid	You might want to employ a combination of Amazon S3 server-side encryption and client-side encryption, as well as application-level encryption.

AWS Partner Network (APN) partners provide specialized solutions for protecting data at rest and in transit on Amazon EMR, for more information visit the AWS Security Partner Solutions page.

Decommission Data and Media Securely

You decommission data differently in the cloud than you do in traditional on- premises environments.

When you ask AWS to delete data in the cloud, AWS does not decommission the underlying physical media; instead, the storage blocks are marked as unallocated. AWS uses secure mechanisms to reassign the blocks elsewhere. When you provision block storage, the hypervisor or virtual machine manager (VMM) keeps track of which blocks your instance has written to. When an instance writes to a block of storage, the previous

block is zeroed out, and then overwritten with your block of data. If your instance attempts to read from a block previously written to, your previously stored data is returned. If an instance attempts to read from a block it has not previously written to, the hypervisor zeros out the previous data on disk and returns a zero to the instance.

When AWS determines that media has reached the end of its useful life, or it experiences a hardware fault, AWS follows the techniques detailed in Department of Defense (DoD) 5220.22-M (“National Industrial Security Program Operating Manual”) or NIST SP 800-88 (“Guidelines for Media Sanitization”) to destroy data as part of the decommissioning process.

For more information about deletion of data in the cloud, see the AWS Overview of Security Processes whitepaper.

When you have regulatory or business reasons to require further controls for securely decommissioning data, you can implement data encryption at rest using customer managed keys, which are not stored in the cloud. Then in addition to following the previous process, you would delete the key used to protect the decommissioned data, making it irrecoverable.

Protect Data in Transit

Cloud applications often communicate over public links, such as the Internet, so it is important to protect data in transit when you run applications in the cloud. This involves protecting network traffic between clients and servers, and network traffic between servers. Table 12 lists common concerns to communication over public links, such as the Internet.

Table 12: Threats to data in transit

Concern	Comments	Recommended Protection
Accidental information disclosure	Access to your confidential data should be limited. When data is traversing the public network, it should be protected from disclosure through encryption.	Encrypt data in transit using IPSec ESP and/or SSL/TLS.

Concern	Comments	Recommended Protection
Data integrity compromise	Whether or not data is confidential, you want to know that data integrity is not compromised through deliberate or accidental modification.	Authenticate data integrity using IPSec ESP/AH, and/or SSL/TLS.
Peer identity compromise/ identity spoofing/ man- in-the-middle	Encryption and data integrity authentication are important for protecting the communications channel. It is equally important to authenticate the identity of the remote end of the connection. An encrypted channel is worthless if the remote end happens to be an attacker, or an imposter relaying the connection to the intended recipient.	Use IPSec with IKE with pre-shared keys or X.509 certificates to authenticate the remote end. Alternatively, use SSL/TLS with server certificate authentication based on the server common name (CN), or Alternative Name (AN/SAN).

Services from AWS provide support for both IPSec and SSL/TLS for protection of data in transit. IPSec is a protocol that extends the IP protocol stack, often in network infrastructure, and allows applications on upper layers to communicate securely without modification. SSL/TLS, on the other hand, operates at the session layer, and while there are third-party SSL/TLS wrappers, it often requires support at the application layer as well.

The following sections provide details on protecting data in transit.

Managing Application and Administrative Access to AWS Public Cloud Services

When accessing applications running in the AWS public cloud, your connections traverse the Internet. In most cases your security policies consider the Internet an insecure communications medium and require application data protection in transit. Table 13 outlines approaches for protecting data in transit when accessing public cloud services.

Table 13: Protecting application data in transit when accessing public cloud

Protocol/Scenario	Description	Recommended Protection Approach
HTTP/HTTPS traffic (web applications)	By default, HTTP traffic is unprotected. SSL/TLS protection for HTTP traffic, also known as HTTPS, is industry standard and widely supported by web servers and browsers. HTTP traffic can include not just client access to web pages but also web services (REST-based access) as well.	Use HTTPS (HTTP over SSL/TLS) with server certificate authentication.
HTTPS offload (web applications)	While using HTTPS is often recommended, especially for sensitive data, SSL/TLS processing requires additional CPU and memory resources from both the web server and the client. This can put a considerable load on web servers handling thousands of SSL/TLS sessions. There is less impact on the client, where only a limited number of SSL/TLS connections are terminated.	Offload HTTPS processing on Elastic Load Balancing to minimize impact on web servers while still protecting data in transit. Further protect the backend connection to instances using an application protocol such as HTTP over SSL.
Remote Desktop Protocol (RDP) traffic	Users who access Windows Terminal Services in the public cloud usually use the Microsoft Remote Desktop Protocol (RDP). By default, RDP connections establish an underlying SSL/TLS connection.	For optimal protection, the Windows server being accessed should be issued a trusted X.509 certificate to protect from identity spoofing or man- in-the-middle attacks. By default, Windows RDP servers use self-signed certificates, which are not trusted, and should be avoided.

Protocol/Scenario	Description	Recommended Protection Approach
Secure Shell (SSH) traffic	SSH is the preferred approach for establishing administrative connections to Linux servers. SSH is a protocol that, like SSL, provides a secure communications channel between the client and the server. In addition, SSH also supports tunneling, which you should use for running applications such as X-Windows on top of SSH, and protecting the application session in transit.	Use SSH version 2 using non-privileged user accounts.
Database server traffic	If clients or servers need to access databases in the cloud, they might need to traverse the Internet as well.	Most modern databases support SSL/TLS wrappers for native database protocols. For database servers running on Amazon EC2, we recommend this approach to protecting data in transit. Amazon RDS provides support for SSL/TLS in some cases. See the Protecting Data in Transit to Amazon RDS section for more details.

Protecting Data in Transit when Managing AWS Services

You can manage your services from AWS, such as Amazon EC2 and Amazon S3, using the AWS Management Console or AWS APIs. Examples of service management traffic include launching a new Amazon EC2 instance, saving an object to an Amazon S3 bucket, or amending a security group on Amazon VPC.

The AWS Management Console uses SSL/TLS between the client browser and console service endpoints to protect AWS service management traffic. Traffic is encrypted, data integrity is authenticated, and the client browser authenticates the identity of the console service endpoint by using an X.509 certificate. After an SSL/TLS session is established between the client browser and the console service endpoint, all subsequent HTTP traffic is protected within the SSL/TLS session.

You can alternatively use AWS APIs to manage services from AWS either directly from applications or third-party tools, or via SDKs, or via AWS command line tools. AWS

APIs are web services (REST) over HTTPS. SSL/TLS sessions are established between the client and the specific AWS service endpoint, depending on the APIs used, and all subsequent traffic, including the REST envelope and user payload, is protected within the SSL/TLS session.

Protecting Data in Transit to Amazon S3

Like AWS service management traffic, Amazon S3 is accessed over HTTPS. This includes all Amazon S3 service management requests as well as user payload, such as the contents of objects being stored/retrieved from Amazon S3, and associated metadata.

When the AWS service console is used to manage Amazon S3, an SSL/TLS secure connection is established between the client browser and the service console endpoint. All subsequent traffic is protected within this connection.

When Amazon S3 APIs are used directly or indirectly, an SSL/TLS connection is established between the client and the Amazon S3 endpoint, and then all subsequent HTTP, and user payload traffic is encapsulated within the protected session.

Protecting Data in Transit to Amazon RDS

If you're connecting to Amazon RDS from Amazon EC2 instances in the same region, you can rely on the security of the AWS network, but if you're connecting from the Internet, you might want to use SSL/TLS for additional protection.

SSL/TLS provides peer authentication via server X.509 certificates, data integrity authentication, and data encryption for the client-server connection.

SSL/TLS is currently supported for connections to Amazon RDS MySQL and Microsoft SQL instances. For both products, Amazon Web Services provides a single self-signed certificate associated with the MySQL or Microsoft SQL listener. You can download the self-signed certificate and designate it as trusted. This provides for peer identity authentication and prevents man-in-the-middle or identity-spoofing attacks on the server side. SSL/TLS provides for native encryption and data integrity authentication of the communications channel between the client and the server. Because the same self-signed certificate is used on all Amazon RDS MySQL instances on AWS, and another single self- signed certificate is used across all Amazon RDS Microsoft SQL instances on AWS, peer identity authentication does not provide for individual instance authentication. If you require individual server authentication via SSL/TLS, you might need to leverage Amazon EC2 and self-managed relational database services.

Amazon RDS for Oracle Native Network Encryption encrypts the data as it moves into and out of the database. With Oracle Native Network Encryption you can encrypt network traffic travelling over Oracle Net Services using industry- standard encryption algorithms such as AES and Triple DES.

Protecting Data in Transit to Amazon DynamoDB

If you're connecting to DynamoDB from other services from AWS in the same region, you can rely on the security of the AWS network, but if you're connecting to DynamoDB across the Internet, you should use HTTP over SSL/TLS (HTTPS) to connect to DynamoDB service endpoints. Avoid using HTTP for access to DynamoDB, and for all connections across the Internet.

Protecting Data in Transit to Amazon EMR

Amazon EMR includes a number of application communication paths, each of which requires separate protection mechanisms for data in transit. Table 14 outlines the communications paths and the protection approach we recommend.

Table 14: Protecting data in transit on Amazon EMR

Type of Amazon EMR Traffic	Description	Recommended Protection Approach
Between Hadoop nodes	Hadoop Master, Worker, and Core nodes all communicate with one another using proprietary plain TCP connections. However, all Hadoop nodes on Amazon EMR reside in the same Availability Zone, and are protected by security standards at the physical and infrastructure layer.	No additional protection typically required– all nodes reside in the same facility.
Between Hadoop Cluster and Amazon S3	Amazon EMR uses HTTPS to send data between DynamoDB and Amazon EC2. For more information, see the Protecting Data in Transit to Amazon S3 section.	HTTPS used by default.

Type of Amazon EMR Traffic	Description	Recommended Protection Approach
Between Hadoop Cluster and Amazon DynamoDB	Amazon EMR uses HTTPS to send data between Amazon S3 and Amazon EC2. For more information, see the Protecting Data in Transit to Amazon DynamoDB section.	HTTPS used by default. Use SSL/TLS if Thrift, REST, or Avro are used.
User or application access to Hadoop cluster	Clients or applications on premises can access Amazon EMR clusters across the Internet using scripts (SSH-based access), REST, or protocols such as Thrift or Avro.	Use SSH for interactive access to applications, or for tunneling other protocols within SSH.
Administrative access to Hadoop cluster	Amazon EMR cluster administrators typically use SSH to manage the cluster.	Use SSH to the Amazon EMR master node.

Secure Your Operating Systems and Applications

With the AWS shared responsibility model, you manage your operating systems and applications security. Amazon EC2 presents a true virtual computing environment, in which you can use web service interfaces to launch instances with a variety of operating systems with custom preloaded applications. You can standardize operating system and application builds and centrally manage the security of your operating systems and applications in a single secure build repository. You can build and test a pre-configured AMI to meet your security requirements.

Recommendations include:

- Disable root API access keys and secret key
- Restrict access to instances from limited IP ranges using Security Groups
- Password protect the .pem file on user machines

- Delete keys from the authorizedkeys file on your instances when someone leaves your organization or no longer requires access
- Rotate credentials (DB, Access Keys)
- Regularly run least privilege checks using IAM user Access Advisor and IAM user Last Used Access Keys
- Use bastion hosts to enforce control and visibility

This section is not intended to provide a comprehensive list of hardening standards for AMIs. Sources of industry- accepted system hardening standards include, but are not limited to:

- Center for Internet Security (CIS)
- International Organization for Standardization (ISO)
- SysAdmin Audit Network Security (SANS) Institute
- National Institute of Standards Technology (NIST)

We recommend that you develop configuration standards for all system components. Ensure that these standards address all known security vulnerabilities and are consistent with industry-accepted system hardening standards.

If a published AMI is found to be in violation of best practices, or poses a significant risk to customers running the AMI, AWS reserves the right to take measures to remove the AMI from the public catalog and notify the publisher and those running the AMI of the findings.

Creating Custom AMIs

You can create your own AMIs that meet the specific requirements of your organization and publish them for internal (private) or external (public) use. As a publisher of an AMI, you are responsible for the initial security posture of the machine images that you use in production. The security controls you apply on the AMI are effective at a specific point in time, they are not dynamic. You can configure private AMIs in any way that meets your business needs and does not violate the AWS Acceptable Use Policy. For more information, see the Amazon Web Services Acceptable Use Policy.

Users who launch from AMIs, however, might not be security experts, so we recommend that you meet certain minimum security standards.

Before you publish an AMI, make sure that the published software is up to date with relevant security patches and perform the clean-up and hardening tasks listed in Table 15.

Table 15: Clean up tasks before publishing an AMI

Area	Recommended Task
Disable insecure applications	Disable services and protocols that authenticate users in clear text over the network, or otherwise insecurely.
Minimize exposure	Disable non-essential network services on startup. Only administrative services (SSH/RDP) and the services required for essential applications should be started.
Protect credentials	Securely delete all AWS credentials from disk and configuration files.
Protect credentials	Securely delete any third-party credentials from disk and configuration files.
Protect credentials	Securely delete all additional certificates or key material from the system.
Protect credentials	Ensure that software installed does not use default internal accounts and passwords.
Use good governance	Ensure that the system does not violate the Amazon Web Services Acceptable Use Policy. Examples of violations include open SMTP relays or proxy servers. For more information, see the Amazon Web Services Acceptable Use Policy

Tables 16 and 17 list additional operating-system specific clean up tasks. Table 16 lists the steps for securing Linux AMIs.

Table 16: Securing Linux/UNIX AMIs

Area	Hardening Activity
Secure services	Configure sshd to allow only public key authentication. Set PubkeyAuthentication to Yes and PasswordAuthentication to No in sshd_config.
Secure services	Generate a unique SSH host key on instance creation. If the AMI uses cloud-init, it will handle this automatically.

Area	Hardening Activity
Protect credentials	Remove and disable passwords for all user accounts so that they cannot be used to log in and do not have a default password. Run passwd -l <USERNAME> for each account.
Protect credentials	Securely delete all user SSH public and private key pairs.
Protect data	Securely delete all shell history and system log files containing sensitive data.

Table 17: Securing Windows AMIs

Area	Hardening Activity
Protect credentials	Ensure that all enabled user accounts have new randomly generated passwords upon instance creation. You can configure the EC2 Config Service to do this for the Administrator account upon boot, but you must explicitly do so before bundling the image.
Protect credentials	Ensure that the Guest account is disabled.
Protect data	Clear the Windows event logs.
Protect credentials	Make sure the AMI is not a part of a Windows domain.
Minimizing exposure	Do not enable any file sharing, print spooler, RPC, and other Windows services that are not essential but are enabled by default.

Bootstrapping

After the hardened AMI is instantiated, you can still amend and update security controls by using bootstrapping applications. Common bootstrapping applications include Puppet, Chef, Capistrano, Cloud-Init and Cfn-Init. You can also run custom bootstrapping Bash or Microsoft Windows PowerShell scripts without using third-party tools.

Here are a few bootstrap actions to consider:

- Security software updates install the latest patches, service packs, and critical updates beyond the patch level of the AMI.

- Initial application patches install application level updates, beyond the current application level build as captured in the AMI.
- Contextual data and configuration enables instances to apply configurations specific to the environment in which they are being launched–production, test, or DMZ/internal, for example.
- Register instances with remote security monitoring and management systems.

Managing Patches

You are responsible for patch management for your AMIs and live instances. We recommend that you institutionalize patch management and maintain a written procedure.

While you can use third-party patch management systems for operating systems and major applications, it is a good practice to keep an inventory of all software and system components, and to compare the list of security patches installed on each system to the most recent vendor security patch list, to verify that current vendor patches are installed.

Implement processes to identify new security vulnerabilities and assign risk rankings to such vulnerabilities. At a minimum, rank the most critical, highest risk vulnerabilities as "High."

Controlling Security for Public AMIs

Take care that you don't leave important credentials on AMIs when you share them publicly. For more information, see How To Share and Use Public AMIs in A Secure Manner.

Protecting Your System from Malware

Protect your systems in the cloud as you would protect a conventional infrastructure from threats such as viruses, worms, Trojans, rootkits, botnets, and spam.

It's important to understand the implications of a malware infection to an individual instance, as well as to the entire cloud system: When a user–wittingly or unwittingly–executes a program on a Linux or Windows system, the executable assumes the privileges of that user (or in some cases, impersonates another user). The code can carry out any action that the user who launched it has permissions for. Users must ensure that they only execute trusted code.

If you execute a piece of untrusted code on your system, it's no longer your system–it belongs to someone else. If a superuser or a user with administrative privileges executes an untrusted program, the system on which the program was executed can no longer be trusted–malicious code might change parts of the operating system, install a rootkit, or establish back doors for accessing the system. It might delete data or compromise data integrity, or compromise the availability of services, or disclose information in a covert or overt fashion to third parties.

Consider the instance on which the code was executed to be infected. If the infected instance is part of a single sign-on environment, or if there is an implicit trust model for access between instances, the infection can quickly spread beyond the individual instance into the entire system and beyond. An infection of this scale can quickly lead to data leakage, data and service compromise, and it can erode the company's reputation. It might also have direct financial consequences, if, for example, it compromises services to third parties or over-consumes cloud resources. You must manage the threat of malware. Table 18 outlines some common approaches to malware protection.

Table 18: Approaches for protection from malware

Factor	Common Approaches
Untrusted AMIs	Launch instances from trusted AMIs only. Trusted AMIs include the standard Windows and Linux AMIs provided by AWS and AMIs from trusted third parties. If you derive your own custom AMIs from the standard and trusted AMIs, all the additional software and settings you apply to it must be trusted as well. Launching an untrusted third-party AMI can compromise and infect your entire cloud environment.
Untrusted software	Only install and run trusted software from a trusted software provider. A trusted software provider is one who is well regarded in the industry, and develops software in a secure and responsible fashion, not allowing malicious code into its software packages. Open source software can also be trusted software, and you should be able to compile your own executables. We strongly recommend that you perform careful code reviews to ensure that source code is non-malicious. Trusted software providers often sign their software using code-signing certificates or provide MD5 or SHA-1 signatures of their products so that you can verify the integrity of the software you download.

Factor	Common Approaches
Untrusted software depots	You download trusted software from trusted sources. Random sources of software on the Internet or elsewhere on the network might actually be distributing malware inside an otherwise legitimate and reputable software package. Such untrusted parties might provide MD5 or SHA-1 signatures of the derivative package with malware in it, so such signatures should not be trusted. We advise that you set up your own internal software depots of trusted software for your users to install and use. Strongly discourage users from the dangerous practice of downloading and installing software from random sources on the Internet.
Principle of least privilege	Give users the minimum privileges they need to carry out their tasks. That way, even if a user accidentally launches an infected executable, the impact on the instance and the wider cloud system is minimized.
Patching	Patch external-facing and internal systems to the latest security level. Worms often spread through unpatched systems on the network.
Botnets	If an infection–whether from a conventional virus, a Trojan, or a worm–spreads beyond the individual instance and infects a wider fleet, it might carry malicious code that creates a botnet–a network of infected hosts that can be controlled by a remote adversary. Follow all the previous recommendations to avoid a botnet infection.
Spam	Infected systems can be used by attackers to send large amounts of unsolicited mail (spam). AWS provides special controls to limit how much email an Amazon EC2 instance can send, but you are still responsible for preventing infection in the first place. Avoid SMTP open relay, which can be used to spread spam, and which might also represent a breach of the AWS Acceptable Use Policy. For more information, see the Amazon Web Services Acceptable Use Policy. .
Antivirus/ Antispam software	Be sure to use a reputable and up-to-date antivirus and antispam solution on your system.
Host-based IDS software	Many AWS customers install host-based IDS software, such as the open source product OSSEC, that includes file integrity checking and rootkit detection software. Use these products to analyze important system files and folders and calculate checksum that reflect their trusted state, and then regularly check to see whether these files have been modified and alert the system administrator if so.

If an instance is infected, antivirus software might be able to detect the infection and remove the virus. We recommend the most secure and widely recommended approach, which is to save all the system data, then reinstall all the systems, platforms, and

application executables from a trusted source, and then restore the data only from backup.

Mitigating Compromise and Abuse

AWS provides a global infrastructure for customers to build solutions on, many of which face the Internet. Our customer solutions must operate in a manner that does no harm to the rest of Internet community, that is, they must avoid abuse activities.

Abuse activities are externally observed behaviors of AWS customers' instances or other resources that are malicious, offensive, illegal, or could harm other Internet sites.

AWS works with you to detect and address suspicious and malicious activities from your AWS resources. Unexpected or suspicious behaviors from your resources can indicate that your AWS resources have been compromised, which signals potential risks to your business. AWS uses the following mechanisms to detect abuse activities from customer resources:

- AWS internal event monitoring
- External security intelligence against AWS network space
- Internet abuse complaints against AWS resources

While the AWS abuse response team aggressively monitors and shuts down malicious abusers or fraudsters running on AWS, the majority of abuse complaints refer to customers who have legitimate business on AWS. Common causes of unintentional abuse activities include:

- **Compromised resource.** For example, an unpatched Amazon EC2 instance could be infected and become a botnet agent.
- **Unintentional abuse.** For example, an overly aggressive web crawler might be classified as a DOS attacker by some Internet sites.
- **Secondary abuse.** For example, an end user of the service provided by an AWS customer might post malware files on a public Amazon S3 bucket.
- **False complaints.** Internet users might mistake legitimate activities for abuse.

AWS is committed to working with AWS customers to prevent, detect, and mitigate abuse, and to defend against future re-occurrences. When you receive an AWS abuse warning, your security and operational staffs must immediately investigate the matter. Delay can prolong the damage to other Internet sites and lead to reputation and legal

liability for you. More importantly, the implicated abuse resource might be compromised by malicious users, and ignoring the compromise could magnify damages to your business.

Malicious, illegal, or harmful activities that use your AWS resources violate the AWS Acceptable Use Policy and can lead to account suspension. For more information, see the Amazon Web Services Acceptable Use Policy. It is your responsibility to maintain a well- behaved service as evaluated by the Internet community. If an AWS customer fails to address reported abuse activities, AWS will suspend the AWS account to protect the integrity of the AWS platform and the Internet community. Table 19 lists best practices that can help you respond to abuse incidents:

Table 19: Best practices for mitigating abuse

Best Practice	Description
Never ignore AWS abuse communication.	When an abuse case is filed, AWS immediately sends an email notification to the customer's registered email address. You can simply reply to the abuse warning email to exchange information with the AWS abuse response team. All communications are saved in the AWS abuse tracking system for future reference. The AWS abuse response team is committed to helping customers to understand the nature of the complaints. AWS helps customers to mitigate and prevent abuse activities. Account suspension is the last action the AWS abuse response team takes to stop abuse activities. We work with our customers to mitigate problems and avoid having to take any punitive action. But you must respond to abuse warnings, take action to stop the malicious activities, and prevent future re-occurrence. Lack of customer response is the leading reason for instance and account blocks.
Follow security best practices.	The best protection against resource compromise is to follow the security best practices outlined in this document. While AWS provides certain security tools to help you establish strong defenses for your cloud environment, you must follow security best practices as you would for servers within your own data center. Consistently adopt simple defense practices, such as applying the latest software patches, restricting network traffic via a firewall and/or Amazon EC2 security groups, and providing least privilege access to users.

Best Practice	Description
Mitigation to compromises.	If your computing environment has been compromised or infected, we recommend taking the following steps to recover to a safe state: Consider any known compromised Amazon EC2 instance or AWS resource unsafe. If your Amazon EC2 instance is generating traffic that cannot be explained by your application usage, your instance has probably been compromised or infected with malicious software. Shut down and rebuild that instance completely to get back to a safe state. While a fresh re-launch can be challenging in the physical world, in the cloud environment it is the first mitigation approach. You might need to carry out forensic analysis on a compromised instance to detect the root cause. Only well-trained security experts should perform such an investigation, and you should isolate the infected instance to prevent further damage and infection during the investigation. To isolate an Amazon EC2 instance for investigation, you can set up a very restrictive security group, for example, close all ports except to accept inbound SSH or RDP traffic from one single IP address from which the forensic investigator can safely examine the instance. You can also take an offline Amazon EBS snapshot of the infected instance and then deliver the offline snapshot to forensic investigators for deep analysis. AWS does not have access to the private information inside your instances or other resources, so we cannot detect guest operating system or application level compromises, such as application account take-over. AWS cannot retroactively provide information (such as access logs, IP traffic logs, or other attributes) if you are not recording that information via your own tools. Most deep incident investigation and mitigation activities are your responsibility. The final step you must take to recover from compromised Amazon EC2 instances is to back up key business data, completely terminate the infected instances, and re-launch them as fresh resources. To avoid future compromises, we recommend that you review the security control environment on the newly launched instances. Simple steps like applying the latest software patches and restricting firewalls go a long way.
Set up security communication email address.	The AWS abuse response team uses email for abuse warning notifications. By default, this email goes to your registered email address, but if you are in a large enterprise, you might want to create a dedicated response email address. You can set up additional email addresses on your Personal Information page, under Configure Additional Contacts.

Using Additional Application Security Practices

Here are some additional general security best practices for your operating systems and applications:

- Always change vendor-supplied defaults before creating new AMIs or prior to deploying new applications, including but not limited to passwords, simple network management protocol (SNMP) community strings, and security configuration.
- Remove or disable unnecessary user accounts.
- Implement a single primary function per Amazon EC2 instance to keep functions that require different security levels from co-existing on the same server. For example, implement web servers, database servers, and DNS on separate servers.
- Enable only necessary and secure services, protocols, daemons, etc., as required for the functioning of the system. Disable all non-essential services, because they increase the security risk exposure for the instance, as well as the entire system.
- Disable or remove all unnecessary functionality, such as scripts, drivers, features, subsystems, EBS volumes

Configure all services with security best practices in mind. Enable security features for any required services, protocols, or daemons. Choose services such as SSH, which have built-in security mechanisms for user/peer authentication, encryption and data integrity authentication, over less secure equivalents such as Telnet. Use SSH for file transfers, rather than insecure protocols like FTP. Where you can't avoid using less secure protocols and services, introduce additional security layers around them, such as IPSec or other virtual private network (VPN) technologies to protect the communications channel at the network layer, or GSS-API, Kerberos, SSL or TLS to protect network traffic at the application layer.

While security governance is important for all organizations, it is a best practice to enforce security policies. Wherever possible, configure your system security parameters to comply with your security policies and guidelines to prevent misuse.

For administrative access to systems and applications, encrypt all non-console administrative access using strong cryptographic mechanisms. Use technologies such

as SSH, user and site-to-site IPSec VPNs, or SSL/TLS to further secure remote system management.

Secure Your Infrastructure

This section provides recommendations for securing infrastructure services on the AWS platform.

Using Amazon Virtual Private Cloud (VPC)

With Amazon Virtual Private Cloud (VPC) you can create private clouds within the AWS public cloud.

Each customer Amazon VPC uses IP address space, allocated by customer. You can use private IP addresses (as recommended by RFC 1918) for your Amazon VPCs, building private clouds and associated networks in the cloud that are not directly routable to the Internet.

Amazon VPC provides not only isolation from other customers in the private cloud, it provides layer 3 (Network Layer IP routing) isolation from the Internet as well. Table 20 lists options for protecting your applications in Amazon VPC:

Table 20: Accessing resources in Amazon VPC

Concern	Description	Recommended Protection Approach
Internet-only	The Amazon VPC is not connected to any of your infrastructure on premises or elsewhere. You might or might not have additional infrastructure residing on premises, or elsewhere. If you need to accept connections from Internet users, you can provide inbound access by allocating elastic IP addresses (EIPs) to only those Amazon VPC instances that need them. You can further limit inbound connections by using security groups or NACLs for only specific ports and source IP address ranges. If you can balance the load of traffic inbound from the Internet, you don't need EIPs. You can place instances behind Elastic Load Balancing. For outbound (to the Internet) access, for example to fetch software updates or to access data on AWS public services, such as Amazon S3, you can use a NAT instance to provide masquerading for outgoing connections. No EIPs are required.	Encrypt application and administrative traffic using SSL/TLS, or build custom user VPN solutions. Carefully plan routing and server placement in public and private subnets. Use security groups and NACLs.
IPSec over the Internet	AWS provides industry-standard and resilient IPSec termination infrastructure for VPC. Customers can establish IPSec tunnels from their on-premises or other VPN infrastructure to Amazon VPC. IPSec tunnels are established between AWS and your infrastructure endpoints. Applications running in the cloud or on premises don't require any modification and can benefit from IPSec data protection in transit immediately.	Establish a private IPSec connection using IKEv1 and IPSec using standard AWS VPN facilities (Amazon VPC VPN gateways, customer gateways, and VPN connections). Alternatively, establish customer- specific VPN software infrastructure in the cloud, and on-premises.
AWS Direct Connect without IPSec	With AWS Direct Connect, you can establish a connection to your Amazon VPC using private peering with AWS over dedicated links, without using the Internet. You can opt to not use IPSec in this case, subject to your data protection requirements.	Depending on your data protection requirements, you might not need additional protection over private peering.

Concern	Description	Recommended Protection Approach
AWS Direct Connect with IPSec	You can use IPSec over AWS Direct Connect links for additional end-to- end protection.	See IPSec over the Internet above.
Hybrid	Consider using a combination of these approaches. Employ adequate protection mechanisms for each connectivity approach you use.	

You can leverage Amazon VPC-IPSec or VPC-AWS Direct Connect to seamlessly integrate on-premises or other hosted infrastructure with your Amazon VPC resources in a secure fashion. With either approach, IPSec connections protect data in transit, while BGP on IPSec or AWS Direct Connect links integrate your Amazon VPC and on-premises routing domains for transparent integration for any application, even applications that don't support native network security mechanisms.

Although VPC-IPSec provides industry-standard and transparent protection for your applications, you might want to use additional levels of protection mechanisms, such as SSL/TLS over VPC-IPSec links.

For more information, please refer to the Amazon VPC Connectivity Options whitepaper.

Using Security Zoning and Network Segmentation

Different security requirements mandate different security controls. It is a security best practice to segment infrastructure into zones that impose similar security controls.

While most of the AWS underlying infrastructure is managed by AWS operations and security teams, you can build your own overlay infrastructure components. Amazon VPCs, subnets, routing tables, segmented/zoned applications and custom service instances such as user repositories, DNS, and time servers supplement the AWS managed cloud infrastructure.

Usually, network engineering teams interpret segmentation as another infrastructure design component and apply network-centric access control and firewall rules to manage access. Security zoning and network segmentation are two different concepts, however: A network segment simply isolates one network from another, where a security zone creates a group of system components with similar security levels with common controls.

On AWS, you can build network segments using the following access control methods:

- Using **Amazon VPC** to define an isolated network for each workload or organizational entity.
- Using **security groups** to manage access to instances that have similar functions and security requirements; security groups are stateful firewalls that enable firewall rules in both directions for every allowed and established TCP session or UDP communications channel.
- Using **Network Access Control Lists** (NACLs) that allow stateless management of IP traffic. NACLs are agnostic of TCP and UDP sessions, but they allow granular control over IP protocols (for example GRE, IPSec ESP, ICMP), as well as control on a per-source/destination IP address and port for TCP and UDP. NACLs work in conjunction with security groups, and can allow or deny traffic even before it reaches the security group.
- Using **host-based firewalls** to control access to each instance.
- Creating a **threat protection layer** in traffic flow and enforcing all traffic to traverse the zone.
- Applying **access control at other layers** (e.g. applications and services).

Traditional environments require separate network segments representing separate broadcast entities to route traffic via a central security enforcement system such as a firewall. The concept of security groups in the AWS cloud makes this requirement obsolete. Security groups are a logical grouping of instances, and they also allow the enforcement of inbound and outbound traffic rules on these instances regardless of the subnet where these instances reside.

Creating a security zone requires additional controls per network segment, and they often include:

- **Shared Access Control**–a central Identity and Access Management (IDAM) system. Note that although federation is possible, this will often be separate from IAM.
- **Shared Audit Logging**–shared logging is required for event analysis and correlation, and tracking security events.
- **Shared Data Classification**–see Table 1: Sample Asset Matrix Design Your ISMS to Protect Your Assets section for more information.

- **Shared Management Infrastructure**–various components, such as anti-virus/antispam systems, patching systems, and performance monitoring systems.
- **Shared Security (Confidentiality/Integrity) Requirements**–often considered in conjunction with data classification.

To assess your network segmentation and security zoning requirements, answer the following questions:

- Do I control inter-zone communication? Can I use network segmentation tools to manage communications between security zones A and B? Usually access control elements such as security groups, ACLs, and network firewalls should build the walls between security zones. Amazon VPCs by default builds inter-zone isolation walls.
- Can I monitor inter-zone communication using an IDS/IPS/DLP/SIEM/NBAD system, depending on business requirements? Blocking access and managing access are different terms. The porous communication between security zones mandates sophisticated security monitoring tools between zones. The horizontal scalability of AWS instances makes it possible to zone each instance at the operating systems level and leverage host-based security monitoring agents.
- Can I apply per zone access control rights? One of the benefits of zoning is controlling egress access. It is technically possible to control access by resources such as Amazon S3 and Amazon SMS resources policies.
- Can I manage each zone using dedicated management channel/roles? Role-Based Access Control for privileged access is a common requirement. You can use IAM to create groups and roles on AWS to create different privilege levels. You can also mimic the same approach with application and system users. One of the new key features of Amazon VPC–based networks is support for multiple elastic network interfaces. Security engineers can create a management overlay network using dual homed instances.
- Can I apply per zone confidentiality and integrity rules? Per zone encryption, data classification, and DRM simply increase the overall security posture. If the security requirements are different per security zone, then the data security requirements must be different as well. And it is always a good policy to use different encryption options with rotating keys on each security zone.

AWS provides flexible security zoning options. Security engineers and architects can leverage the following AWS features to build isolated security zones/segments on AWS per Amazon VPC access control:

- Per subnet access control
- Per security group access control
- Per instance access control (host-based)
- Per Amazon VPC routing block
- Per resource policies (S3/SNS/SMS)
- Per zone IAM policies
- Per zone log management
- Per zone IAM users, administrative users
- Per zone log feed
- Per zone administrative channels (roles, interfaces, management consoles)
- Per zone AMIs
- Per zone data storage resources (Amazon S3 buckets or Glacier archives)
- Per zone user directories
- Per zone applications/application controls

With elastic cloud infrastructure and automated deployment, you can apply the same security controls across all AWS regions. Repeatable and uniform deployments improve your overall security posture.

Strengthening Network Security

Following the shared responsibility model, AWS configures infrastructure components such as data center networks, routers, switches, and firewalls in a secure fashion. You are responsible for controlling access to your systems in the cloud and for configuring network security within your Amazon VPC, as well as secure inbound and outbound network traffic.

While applying authentication and authorization for resource access is essential, it doesn't prevent adversaries from acquiring network-level access and trying to impersonate authorized users. Controlling access to applications and services based on network locations of the user provides an additional layer of security. For example, a web-based application with strong user authentication could also benefit from an IP-address based firewall that limits source traffic to a specific range of IP addresses, and

an intrusion prevention system to limit security exposure and minimize the potential attack surface for the application.

Best practices for network security in the AWS cloud include the following:

- Always use security groups: They provide stateful firewalls for Amazon EC2 instances at the hypervisor level. You can apply multiple security groups to a single instance, and to a single ENI.
- Augment security groups with Network ACLs: They are stateless but they provide fast and efficient controls. Network ACLs are not instance- specific so they can provide another layer of control in addition to security groups. You can apply separation of duties to ACLs management and security group management.
- Use IPSec or AWS Direct Connect for trusted connections to other sites. Use Virtual Gateway (VGW) where Amazon VPC-based resources require remote network connectivity.
- Protect data in transit to ensure the confidentiality and integrity of data, as well as the identities of the communicating parties.
- For large-scale deployments, design network security in layers. Instead of creating a single layer of network security protection, apply network security at external, DMZ, and internal layers.
- VPC Flow Logs provides further visibility as it enables you to capture information about the IP traffic going to and from network interfaces in your VPC.

Many of the AWS service endpoints that you interact with do not provide for native firewall functionality or access control lists. AWS monitors and protects these endpoints with state-of-the-art network and application level control systems. You can use IAM policies to restrict access to your resources based on the source IP address of the request.

Securing Periphery Systems: User Repositories, DNS, NTP

Overlay security controls are effective only on top of a secure infrastructure. DNS query traffic is a good example of this type of control. When DNS systems are not properly secured, DNS client traffic can be intercepted and DNS names in queries and responses can be spoofed. Spoofing is a simple but efficient attack against an infrastructure that lacks basic controls. SSL/TLS can provide additional protection.

Some AWS customers use Amazon Route 53, which is a secure DNS service. If you require internal DNS, you can implement a custom DNS solution on Amazon EC2 instances. DNS is an essential part of the solution infrastructure, and as such becomes a critical part of your security management plan. All DNS systems, as well as other important custom infrastructure components, should apply the following controls:

Table 21: Controls for periphery system

Common Control	Description
Separate administrative level access	Implement role separation and access controls to limit access to such services, often separate from access control required for application access, or access to other parts of the infrastructure.
Monitoring, alerting, audit trail	Log and monitor authorized and unauthorized activity.
Network layer access control	Restrict network access to only systems that require it. If possible, apply protocol enforcement for all network level access attempts (that is, enforce custom RFC standards for NTP and DNS).
Latest stable software with security patches	Ensure that the software is patched and not subject to any known vulnerabilities or other risks.
Continuous security testing (assessments)	Ensure that the infrastructure is tested regularly.
All other security controls processes in place	Make sure the periphery systems follow your information security management system (ISMS) best practices, in addition to service-specific custom security controls.

In addition to DNS, other infrastructure services might require specific controls.

Centralized access control is essential for managing risk. The IAM service provides role-based identity and access management for AWS, but AWS does not provide end-user repositories like Active Directory, LDAP, or RADIUS for your operating systems and applications. Instead, you establish user identification and authentication systems, alongside Authentication Authorization Accounting (AAA) servers, or sometimes proprietary database tables. All identity and access management servers for the purposes of user platforms and applications are critical to security and require special attention.

Time servers are also critical custom services. They are essential in many security-related transactions, including log time-stamping and certificate validation. It is important to use a centralized time server and synchronize all systems with the same time server. The Payment Card Industry (PCI) Data Security Standard (DSS) proposes a good approach to time synchronization:

- Verify that time-synchronization technology is implemented and kept current.
- Obtain and review the process for acquiring, distributing, and storing the correct time within the organization, and review the time-related system- parameter settings for a sample of system components.
- Verify that only designated central time servers receive time signals from external sources, and that time signals from external sources are based on International Atomic Time or Universal Coordinated Time (UTC).
- Verify that the designated central time servers peer with each other to keep accurate time, and that other internal servers receive time only from the central time servers.
- Review system configurations and time-synchronization settings to verify that access to time data is restricted to only personnel who have a business need to access time data.
- Review system configurations and time synchronization settings and processes to verify that any changes to time settings on critical systems are logged, monitored, and reviewed.
- Verify that the time servers accept time updates from specific, industry- accepted external sources. (This helps prevent a malicious individual from changing the clock). (You have the option of receiving those updates encrypted with a symmetric key, and you can create access control lists that specify the IP addresses of client machines that will be updated. (This prevents unauthorized use of internal time servers).

Validating the security of custom infrastructure is an integral part of managing security in the cloud.

Building Threat Protection Layers

Many organizations consider layered security to be a best practice for protecting network infrastructure. In the cloud, you can use a combination of Amazon VPC, implicit firewall rules at the hypervisor-layer, alongside network access control lists, security

groups, host-based firewalls, and IDS/IPS systems to create a layered solution for network security.

While security groups, NACLs and host-based firewalls meet the needs of many customers, if you're looking for defense in-depth, you should deploy a network- level security control appliance, and you should do so inline, where traffic is intercepted and analyzed prior to being forwarded to its final destination, such as an application server.

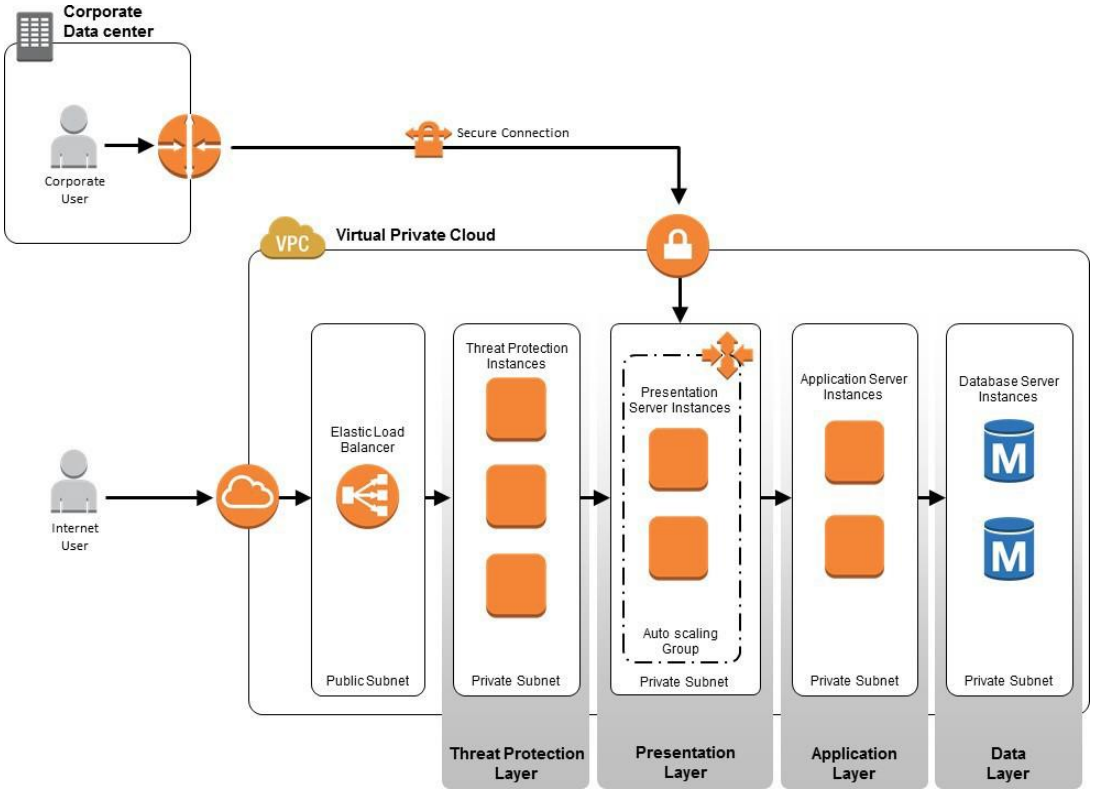

Figure 6: Layered Network Defense in the Cloud

Examples of inline threat protection technologies include the following:

- Third-party firewall devices installed on Amazon EC2 instances (also known as soft blades)
- Unified threat management (UTM) gateways
- Intrusion prevention systems
- Data loss management gateways
- Anomaly detection gateways
- Advanced persistent threat detection gateways

The following key features in the Amazon VPC infrastructure support deploying threat protection layer technologies:

- **Support for Multiple Layers of Load Balancers:** When you use threat protection gateways to secure clusters of web servers, application servers, or other critical servers, scalability is a key issue. AWS reference architectures underline deployment of external and internal load balancers for threat management and internal server load distribution and high availability. You can leverage Elastic Load Balancing or your custom load balancer instances for your multi-tiered designs. You must manage session persistence at the load balancer level for stateful gateway deployments.
- **Support for Multiple IP Addresses:** When threat protection gateways protect a presentation layer that consists of several instances (for example web servers, email servers, application servers), these multiple instances must use one security gateway in a many-to-one relationship. AWS provides support for multiple IP addresses for a single network interface.
- **Support for Multiple Elastic Network Interfaces (ENIs):** Threat protection gateways must be dual-homed and, in many cases, depending on the complexity of the network, must have multiple interfaces. Usingthe concept of ENIs, AWS supports multiple network interfaces on several different instance types, which makes it possible to deploy multi-zone security features.

Latency, complexity, and other architectural constraints sometimes rule out implementing an inline threat management layer, in which case you can choose one of the following alternatives.

- A **distributed threat protection solution**: This approach installs threat protection agents on individual instances in the cloud. A central threat management server communicates with all host-based threat management agents for log collection, analysis, correlation, and active threat response purposes.
- An **overlay network threat protection solution**: Build an overlay network on top of your Amazon VPC using technologies such as GRE tunnels, vtun interfaces, or by forwarding traffic on another ENI to a centralized network traffic analysis and intrusion detection system, which can provide active or passive threat response.

Test Security

Every ISMS must ensure regular reviews of the effectiveness of security controls and policies. To guarantee the efficiency of controls against new threats and vulnerabilities, customers need to ensure that the infrastructure is protected against attacks.

Verifying existing controls requires testing. AWS customers should undertake a number of test approaches:

- **External Vulnerability Assessment:** A third party evaluates system vulnerabilities with little or no knowledge of the infrastructure and its components;
- **External Penetration Tests:** A third party with little or no knowledge of the system actively tries to break into it, in a controlled fashion.
- **Internal Gray/White-box Review of Applications and Platforms**: A tester who has some or full knowledge of the system validates the efficiency of controls in place, or evaluates applications and platforms for known vulnerabilities.

The AWS Acceptable Use Policy outlines permitted and prohibited behavior in the AWS cloud, and defines security violations and network abuse. AWS supports both inbound and outbound penetration testing in the cloud, although you must request permission to conduct penetration tests. For more information, see the Amazon Web Services Acceptable Use Policy.

To request penetration testing for your resources, complete and submit the AWS Vulnerability Penetration Testing Request Form. You must be logged into the AWS Management Console using the credentials associated with the instances you want to test, or the form will not pre-populate correctly. For third-party penetration testing, you must fill out the form yourself and then notify the third parties when AWS grants approval.

The form includes information about the instances to be tested, the expected start and end dates and times of the tests, and requires you to read and agree to the terms and conditions specific to penetration testing and to the use of appropriate tools for testing. AWS policy does not permit testing of m1.small or t1.micro instance types. After you submit the form, you will receive a response confirming receipt of the request within one business day.

If you need more time for additional testing, you can reply to the authorization email asking to extend the test period. Each request is subject to a separate approval process.

Managing Metrics and Improvement

Measuring control effectiveness is an integral process to each ISMS. Metrics provide visibility into how well controls are protecting the environment. Risk management often depends on qualitative and quantitative metrics. Table 22 outlines measurement and improvement best practices:

Table 22: Measuring and improving metrics

Best Practice	Improvement
Monitoring and reviewing procedures and other controls	• Promptly detect errors in the results of processing • Promptly identify attempted and successful security breaches and incidents • Enable management to determine whether the security activities delegated to people or implemented by information technology are performing as expected • Help detect security events and thereby prevent security incidents by the use of indicators • Determine whether the actions taken to resolve a breach of security were effective
Regular reviews of the effectiveness of the ISMS	• Consider results from security audits, incidents, and effectiveness measurements; and suggestions and feedback from all interested parties • Ensure that the ISMS meets the policy and objectives • Review security controls
Measure controls effectiveness	• Verify that security requirements have been met
Risk assessments reviews at planned intervals	• Review the residual risks and the identified acceptable levels of risks, taking into account: • Changes to the organization, technology, business objectives and processes, identified threats • Effectiveness of the implemented controls • External events, such as changes to the legal or regulatory environment, changed contractual obligations, and changes in social climate
Internal ISMS audits	• First party audits (internal audits), are conducted by, or on behalf of, the organization itself for internal purposes.

Best Practice	Improvement
Regular management reviews	• Ensure that the scope remains adequate • Identify improvements in the ISMS process
Update security plans	• Take into account the findings of monitoring and reviewing activities • Record actions and events that could affect the ISMS effectiveness or performance

Mitigating and Protecting Against DoS & DDoS Attacks

Organizations running Internet applications recognize the risk of being the subject of Denial of Service (DoS) or Distributed Denial of Service (DDoS) attacks by competitors, activists, or individuals. Risk profiles vary depending on the nature of business, recent events, the political situation, as well as technology exposure. Mitigation and protection techniques are similar to those used on premises.

If you're concerned about DoS/DDoS attack protection and mitigation, we strongly advise you to enroll in AWS Premium Support services, so that you can proactively and reactively involve AWS support services in the process of mitigating attacks, or containing ongoing incidents in your environment on AWS.

Some services, such as Amazon S3, use a shared infrastructure, which means that multiple AWS accounts access and store data on the same set of Amazon S3 infrastructure components. In this case, a DoS/DDoS attack on abstracted services is likely to affect multiple customers. AWS provides both mitigation and protection controls for DoS/DDoS on abstracted services from AWS to minimize the impact to you in the event such an attack. You are not required to provide additional DoS/DDoS protection of such services, but we do advise that you follow best practices outlined in this whitepaper.

Other services, such as Amazon EC2, use shared physical infrastructure, but you are expected to manage the operating system, platform, and customer data. For such services, we need to work together to provide for effective DDoS mitigation and protection.

AWS uses proprietary techniques to mitigate and contain DoS/DDoS attacks to the AWS platform. To avoid interference with actual user traffic though, and following the shared responsibility model, AWS does not provide mitigation or actively block network

traffic affecting individual Amazon EC2 instances: only you can determine whether excessive traffic is expected and benign, or part of a DoS/DDoS attack.

While a number of techniques can be used to mitigate DoS/DDoS attacks in the cloud, we strongly recommend that you establish a security and performance baseline that captures system parameters under normal circumstances, potentially also considering daily, weekly, annual, or other patterns applicable to your business. Some DoS/DDoS protection techniques, such as statistical and behavioral models, can detect anomalies compared to a given baseline normal operation pattern. For example, a customer who typically expects 2,000 concurrent sessions to their website at a specific time of day might trigger an alarm using Amazon CloudWatch and Amazon SNS if the current number of concurrent sessions exceeds twice that amount (4,000).

Consider the same components that apply to on premises deployments when you establish your secure presence in the cloud. Table 23 outlines common approaches for DoS/DDoS mitigation and protection in the cloud.

Table 23: Techniques for mitigation and protection from DoS/DDoS attacks

Technique	Description	Protection from DoS/DDoS Attacks
Firewalls: Security groups, network access control lists, and host-based firewalls	Traditional firewall techniques limit the attack surface for potential attackers and deny traffic to and from the source of destination of attack.	• Manage the list of allowed destination servers and services (IP addresses & TCP/UDP ports) • Manage the list of allowed sources of traffic protocols • Explicitly deny access temporarily or permanently from specific IP addresses • Manage the list of allowed
Web application firewalls (WAF)	Web application firewalls provide deep packet inspection for web traffic.	• Platform- and application-specific attacks • Protocol sanity attacks • Unauthorized user access
Host-based or inline IDS/IPS systems	IDS/IPS systems can use statistical/behavioral or signature-based algorithms to detect and contain network attacks and Trojans.	• All types of attacks

Technique	Description	Protection from DoS/DDoS Attacks
Traffic shaping/rate limiting	Often DoS/DDoS attacks deplete network and system resources. Rate-limiting is a good technique for protecting scarce resources from overconsumption.	• ICMP flooding • Application request flooding
Embryonic session limits	TCP SYN flooding attacks can take place in both simple and distributed form. In either case, if you have a baseline of the system, you can detect considerable deviations from the norm in the number of half-open (embryonic) TCP sessions, and drop any further TCP SYN packets from the specific sources.	TCP SYN flooding

Along with conventional approaches for DoS/DDoS attack mitigation and protection, the AWS cloud provides capabilities based on its elasticity. DoS/DDoS attacks are attempts to deplete limited compute, memory, disk, or network resources, which often works against on-premises infrastructure. By definition, however, the AWS cloud is elastic, in the sense that new resources can be employed on demand, if and when required. For example, you might be under a DDoS attack from a botnet that generates hundreds of thousands of requests per second that are indistinguishable from legitimate user requests to your web servers. Using conventional containment techniques, you would start denying traffic from specific sources, often entire geographies, on the assumption that there are only attackers and no valid customers there. But these assumptions and actions result in a denial of service to your customers themselves.

In the cloud, you have the option of absorbing such an attack. Using AWS technologies like Elastic Load Balancing and Auto Scaling, you can configure the web servers to scale out when under attack (based on load), and shrink back when the attack stops. Even under heavy attack, the web servers could scale to perform and provide optimal user experience by leveraging cloud elasticity. By absorbing the attack, you might incur additional AWS service costs; but sustaining such an attack is so financially challenging for the attacker that absorbed attacks are unlikely to persist.

You could also use Amazon CloudFront to absorb DoS/DDoS flooding attacks. AWS WAF integrates with AWS CloudFront to help protect your web applications from

common web exploits that could affect application availability, compromise security, or consume excessive resources. Potential attackers trying to attack content behind CloudFront are likely to send most or all requests to CloudFront edge locations where the AWS infrastructure would absorb the extra requests with minimal to no impact on the back-end customer web servers. Again, there would be additional AWS service charges for absorbing the attack, but you should weigh this against the costs the attacker would incur in order to continue the attack as well.

In order to effectively mitigate, contain, and generally manage your exposure to DoS/DDoS attacks, you should build a layer defense model, as outlined elsewhere in this document.

Manage Security Monitoring, Alerting, Audit Trail, and Incident Response

The shared responsibility model requires you to monitor and manage your environment at the operating system and higher layers. You probably already do this on premises or in other environments, so you can adapt your existing processes, tools, and methodologies for use in the cloud.

For extensive guidance on security monitoring, see the ENISA Procure Secure whitepaper, which outlines the concepts of continuous security monitoring in the cloud (see Further Reading).

Security monitoring starts with answering the following questions:

- What parameters should we measure?
- How should we measure them?
- What are the thresholds for these parameters?
- How will escalation processes work?
- Where will data be kept?

Perhaps the most important question you must answer is “What do I need to log?” We recommend configuring the following areas for logging and analysis:

- Actions taken by any individual with root or administrative privileges
- Access to all audit trails

- Invalid logical access attempts
- Use of identification and authentication mechanisms
- Initialization of audit logs
- Creation and deletion of system level objects

When you design your log file, keep the considerations in Table 24 in mind:

Table 24: Log file considerations

Area	Consideration
Log collection	Note how log files are collected. Often operating system, application, or third-party/middleware agents collect log file information.
Log transport	When log files are centralized, transfer them to the central location in a secure, reliable, and timely fashion.
Log storage	Centralize log files from multiple instances to facilitate retention policies, as well as analysis and correlation.
Log taxonomy	Present different categories of log files in a format suitable for analysis.
Log analysis/ correlation	Log files provide security intelligence after you analyze them and correlate events in them. You can analyze logs in real time, or at scheduled intervals.
Log protection/ security	Log files are sensitive. Protect them through network control, identity and access management, encryption, data integrity authentication, and tamper-proof time-stamping.

You might have multiple sources of security logs. Various network components such as firewalls, IDP, DLP, AV systems, the operating system, platforms, and applications will generate log files. Many will be related to security, and those need to be part of the log file strategy. Others, which are not related to security, are better excluded from the strategy. Logs should include all user activities, exceptions, and security events, and you should keep them for a predetermined time for future investigations.

To determine which log files to include, answer the following questions:

- Who are the users of the cloud systems? How do they register, how do they authenticate, how are they authorized to access resources?
- Which applications access cloud systems? How do they get credentials, how do they authenticate, and how they are authorized for such access?

- Which users have privileged access (administrative level access) to AWS infrastructure, operating systems, and applications? How do they authenticate, how are they authorized for such access?

Many services provide built-in access control audit trails (for example, Amazon S3 and Amazon EMR provide such logs) but in some cases, your business requirements for logging might be higher than what's available from the native service log. In such cases, consider using a privilege escalation gateway to manage access control logs and authorization.

When you use a privilege escalation gateway, you centralize all access to the system via a single (clustered) gateway. Instead of making direct calls to the AWS infrastructure, your operating systems or applications, all requests are performed by proxy systems that act as trusted intermediaries to the infrastructure. Often such systems are required to provide or do the following:

- **Automated password management** for privileged access: Privileged access control systems can rotate passwords and credentials based on given policies automatically using built-in connectors for Microsoft Active Directory, UNIX, LDAP, MYSQL, etc.
- **Regularly run least privilege checks** using AWS IAM user Access Advisor and AWS IAM user Last Used Access Keys
- **User authentication** on the front end and delegated access to services from AWS on the back end: Typically a website that provides single sign- on for all users. Users are assigned access privileges based on their authorization profiles. A common approach is using token-based authentication for the website and acquiring click-through access to other systems allowed in the user's profile.
- **Tamper-proof audit trail** storage of all critical activities.
- **Different sign-on credentials for shared accounts:** Sometimes multiple users need to share the same password. A privilege escalation gateway can allow remote access without disclosing the shared account.
- **Restrict leapfrogging or remote desktop hopping** by allowing access only to target systems.
- **Manage commands** that can be used during sessions. For interactive sessions like SSH or appliance management, or AWS CLI, such solutions can enforce policies by limiting the range of available commands and actions.

- Provide **audit trail for terminals and GUI-based sessions** for compliance and security-related purposes.
- **Log everything** and alert based on given threshold for the policies.

Using Change Management Logs

By managing security logs, you can also track changes. These might include planned changes, which are part of the organization's change control process (sometimes referred to as MACD–Move/Add/Change/Delete), ad hoc changes, or unexpected changes, such as incidents. Changes might occur on the infrastructure side of the system, but they might also be related to other categories, such as changes in code repositories, gold image/application inventory changes, process and policy changes, or documentation changes. As a best practice, we recommend employing a tamper-proof log repository for all the above categories of changes. Correlate and interconnect change management and log management systems.

You need a dedicated user with privileges for deleting or modifying change logs; for most systems, devices and applications, change logs should be tamper-proof and regular users should not have privileges to manage the logs. Regular users should be unable to erase evidence from change logs. AWS customers sometimes use file integrity monitoring or change detection software on logs to ensure that existing log data cannot be changed without generating alerts, while adding new entries does not generate alerts.

All logs for system components must be reviewed at the minimum on a daily basis. Log reviews must include those servers that perform security functions, such as intrusion-detection system (IDS) and authentication, authorization, and accounting protocol (AAA) servers (for example, RADIUS). To facilitate this process, you can use log harvesting, parsing, and alerting tools.

Managing Logs for Critical Transactions

For critical applications, all Add, Change/Modify, and Delete activities or transactions must generate a log entry. Each log entry should contain the following information:

- User identification information
- Type of event
- Date and time stamp

- Success or failure indication
- Origination of event
- Identity or name of affected data, system component, or resource

Protecting Log Information

Logging facilities and log information must be protected against tampering and unauthorized access. Administrator and operator logs are often targets for erasing trails of activities.

Common controls for protecting log information include the following:

- Verifying that audit trails are enabled and active for system components
- Ensuring that only individuals who have a job-related need can view audit trail files
- Confirming that current audit trail files are protected from unauthorized modifications via access control mechanisms, physical segregation, and/or network segregation
- Ensuring that current audit trail files are promptly backed up to a centralized log server or media that is difficult to alter
- Verifying that logs for external-facing technologies (for example, wireless, firewalls, DNS, mail) are offloaded or copied onto a secure centralized internal log server or media
- Using file integrity monitoring or change detection software for logs by examining system settings and monitored files and results from monitoring activities
- Obtaining and examining security policies and procedures to verify that they include procedures to review security logs at least daily and that follow-up to exceptions is required
- Verifying that regular log reviews are performed for all system components
- Ensuring that security policies and procedures include audit log retention policies and require audit log retention for a period of time, defined by the business and compliance requirements

Logging Faults

In addition to monitoring MACD events, monitor software or component failure. Faults might be the result of hardware or software failure, and while they might have service and data availability implications, they might not be related to a security incident. Or, a service failure might be the result of deliberate malicious activity, such as a denial of service attack. In any case, faults should generate alerts, and then you should use event analysis and correlation techniques to determine the cause of the fault, and whether it should trigger a security response.

Conclusion

AWS Cloud Platform provides a number of important benefits to modern businesses, including flexibility, elasticity, utility billing, and reduced time-to- market. It provides a range of security services and features that that you can use to manage security of your assets and data in the AWS. While AWS provides an excellent service management layer around infrastructure or platform services, businesses are still responsible for protecting the confidentiality, integrity, and availability of their data in the cloud, and for meeting specific business requirements for information protection.

Conventional security and compliance concepts still apply in the cloud. Using the various best practices highlighted in this whitepaper, we encourage you to build a set of security policies and processes for your organization so you can deploy applications and data quickly and securely.

Contributors

Contributors to this document include:

- Dob Todorov
- Yinal Ozkan

Further Reading

For additional information, see:

- Amazon Web Services: Overview of Security Processes
- Amazon Web Services Risk and Compliance Whitepaper

- Amazon VPC Network Connectivity Options
- AWS SDK support for Amazon S3 client-side encryption
- Amazon S3 Default Encryption for S3 Buckets
- AWS Security Partner Solutions
- Identity Federation Sample Application for an Active Directory Use Case
- Single Sign-on to Amazon EC2 .NET Applications from an On-Premises Windows Domain
- Authenticating Users of AWS Mobile Applications with a Token Vending Machine
- Client-Side Data Encryption with the AWS SDK for Java and Amazon S3
- Amazon Web Services Acceptable Use Policy
- ENISA Procure Secure: A Guide to Monitoring of Security Service Levels in Cloud Contracts
- The PCI Data Security Standard
- ISO/IEC 27001:2013

Document Revisions

Date	Description
August 2016	First publication.