Security Pillar

AWS Well-Architected Framework

May 2017

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

Abstract

The focus of this paper is the Security pillar of the Well-Architected Framework. It provides guidance to help customers apply best practices in the design, delivery, and maintenance of secure AWS environments.

Introduction

At Amazon Web Services (AWS), we understand the value of educating our customers about architectural best practices for designing and operating reliable, secure, efficient, and cost-effective systems in the cloud. As part of this effort, we developed the AWS Well-Architected Framework[1], which helps you understand the pros and cons of decisions you make while building systems on AWS. We believe well-architected systems greatly increase the likelihood of business success.

The framework is based on five pillars:

- Security
- Reliability
- Performance Efficiency
- Cost Optimization
- Operational Excellence

This paper focuses on the Security pillar and how to apply it to your solutions. Ensuring security can be challenging in traditional on-premises solutions due to the use of manual processes, eggshell security models, and insufficient auditing. By adopting the practices in this paper you can build architectures that protect data and systems, control access, and respond automatically to security events.

This paper is intended for those in technology roles, such as chief technology officers (CTOs), architects, developers, and operations team members. After reading this paper, you will understand AWS best practices and strategies to use when designing cloud architectures for Security. This paper doesn't provide implementation details or architectural patterns; however, it does include references to appropriate resources for this information.

Security

The Security pillar encompasses the ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies. This paper will provide in-depth, best-practice guidance for architecting secure systems on AWS.

Design Principles

In the cloud, there are a number of principles that can help you strengthen your system security:

- **Apply security at all layers:** Rather than just running security appliances (e.g., firewalls) at the edge of your infrastructure, use firewalls and other security controls on all of your resources (e.g., every virtual server, load balancer, and network subnet).
- **Enable traceability:** Log and audit all actions and changes to your environment.
- **Implement a principle of least privilege:** Ensure that authorization is appropriate for each interaction with your AWS resources and implement strong logical access controls directly on resources.
- **Focus on securing your system:** With the AWS Shared Responsibility Model[2] you can focus on securing your application, data, and operating systems, while AWS provides secure infrastructure and services.
- **Automate security best practices:** Software-based security mechanisms improve your ability to securely scale more rapidly and cost-effectively. Create and save a patched, hardened image of a virtual server, and then use that image automatically on each new server you launch. Create an entire trust zone architecture that is defined and managed in a template using revision control. Automate the response to both routine and anomalous security events.

Definition

Security in the cloud is composed of five areas:

1. Identity and access management
2. Detective controls
3. Infrastructure protection
4. Data protection
5. Incident response

The AWS Shared Responsibility Model enables organizations that adopt the cloud to achieve their security and compliance goals. Because AWS physically secures the infrastructure that supports our cloud services, AWS customers can focus on using services to accomplish their goals. The AWS Cloud also provides greater access to security data and an automated approach to responding to security events.

Identity and Access Management

Identity and access management are key parts of an information security program, ensuring that only authorized and authenticated users are able to access your resources, and only in a manner that you intend. For example, you should define principals (users, groups, services, and roles that take action in your account), build out policies aligned with these principals, and implement strong credential management. These privilege-management elements form the core of authentication and authorization.

In AWS, there are a number of different approaches to consider when addressing identity and access management, the following sections describe how to use these approaches:

- Protecting AWS credentials
- Fine-grained authorization

Protecting AWS Credentials

The careful management of access credentials is the foundation of how you will secure your resources in the cloud. Because every interaction you make with AWS will be authenticated, establishing appropriate credential management practices and patterns allows you to tie the use of AWS to your workforce lifecycle and ensure that only the appropriate parties take action in your account.

When you open an AWS account, the identity you begin with has access to all AWS services and resources in that account. You use this identity to establish less-privileged users or role-based access in the AWS Identity and Access Management (IAM) service. However, this initial account (known as the *root user*) isn't intended for everyday tasks, and these credentials should be carefully protected using multi-factor authentication (MFA) and by deleting any access keys upon completion of the initial account setup.

For the root account, you should follow the best practice of only using the root account to create another, initial set of IAM users and groups for longer-term identity management operations. These privileged IAM users – carefully monitored and constrained – are used to establish trust with existing identity providers using federation (via SAML 2.0 or web identities) already tied to your organization's workforce source of record. Using federation reduces the need to create users in IAM, while leveraging the existing identities, credentials, and role-based access you might already have established in your organization.

For all IAM users, you should apply appropriate policies enforcing the use of strong authentication. You can set a password policy on the AWS account that requires a minimum length and complexity for passwords associated with IAM users. You can also set a mandatory rotation policy requiring IAM users to change their passwords at regular intervals. For all IAM users with passwords permitting access to the AWS Management Console, you should also require the use of multi-factor authentication.

IAM users might also require access to the AWS APIs directly from command-line tools (CLI) or by using software development kits (SDKs). In these cases, where federation might not be practical, an access key ID and secret access key should be issued and used in place of a password. These credentials should be carefully protected and exchanged for temporary credentials whenever possible. Take extra care to avoid storing access and secret keys in improperly secured locations or inadvertently committing them to source code repositories.

For use cases where federation might not be practical, such as many service-to-service authentication scenarios, you can use IAM instance profiles for Amazon EC2 instances and/or the AWS Security Token Service (STS) to generate and manage temporary credentials used in software that must authenticate to AWS APIs.

Key AWS Services

The key AWS service that supports the protection of credentials is AWS Identity and Access Management (IAM). This service allows you to manage credentials and the policies applied to them. The following services and features are also important:

- **AWS Security Token Service** lets you request temporary, limited-privilege credentials for authentication with other AWS APIs.
- **IAM instance profiles for EC2 instances** allow you to leverage the Amazon Elastic Compute Cloud (EC2) metadata service and managed, temporary credentials for accessing other AWS APIs.

Resources

Refer to the following resources to learn more about AWS best practices for protecting your AWS credentials.

Video

- AWS re:Invent 2015 SEC202 IAM Best Practices to Live By[3]
- AWS re:Invent 2015 SEC307 A Progressive Journey Through AWS IAM Federation Options: From Roles to SAML to Custom Identity Brokers [4]

Documentation

- Root Account Credentials vs. IAM User Credentials[5]
- The Account Root User [6]
- Setting an Account Password Policy for IAM Users
- Managing Access Keys for IAM Users[7]
- Using Instance Profiles[8]
- Temporary Security Credentials[9]

Fine-Grained Authorization

Establishing a principle of least privilege ensures that authenticated identities are only permitted to perform the most minimal set of functions necessary to fulfill a specific task, while balancing usability and efficiency. Operating to this principle limits the *blast radius* - or potential impact - of inappropriate use of

valid credentials, allows you enforce separation of duties for oversight and governance, and makes auditing the entitlements to your resources much simpler.

Organizations should define roles and responsibilities for users and applications interacting with the AWS services and implement fine-grained authorization for enforcement of these roles.

Fine-grained authorization is implemented in AWS using IAM roles and policies. A role is another IAM "principal" assumed by a user or another AWS service and is assigned temporary credentials scoped to a limited set of permissions. IAM policies are documents that formally state one or more permissions. Policies are attached to users, groups, and roles to create a very robust access management framework.

Key AWS Services

The key AWS service supporting fine-grained authorization is AWS Identity and Access Management, which offers a highly flexible policy language used to allow or deny actions, set conditions on these actions, and constrain actions to specific principals or resources.

Resources

Refer to the following resources to learn more about AWS best practices for fine-grained authorization.

Documentation

- Working with Policies[10]
- Delegating Permissions to Administrator IAM Users, Groups, and Credentials[11]
- Managed Policies and Inline Policies[12]
- Working with Policies[13]

Detective Controls

You can use detective controls to identify a potential security incident. They are an essential part of governance frameworks, and can be used to support a quality process, a legal or compliance obligation, and threat identification and response efforts. There are different types of detective controls. For example, conducting an inventory of assets and their detailed attributes promotes more effective decision making (and lifecycle controls) to help establish operational baselines. Or you can use internal auditing, an examination of controls related to information systems, to ensure that practices meet policies and requirements, and that you have set the correct automated alerting notifications based on defined conditions. These controls are important reactive factors that help organizations identify and understand the scope of anomalous activity.

In AWS, there are a number of approaches to consider when addressing detective controls. The following sections describe how to use these approaches:

- Capturing and analyzing logs
- Integrate auditing controls with notification and workflow

Capturing and Analyzing Logs

In traditional data center architectures, aggregating and analyzing logs typically requires installing agents on servers, carefully configuring network appliances to direct log messages at collection points, and forwarding application logs to search and rules engines. Aggregation in the cloud is much easier due to two capabilities.

First, asset management is easier because assets and instances can be described programmatically without depending on agent health. For example, instead of manually updating an asset database and reconciling it with the real install base, you can reliably gather asset metadata with just a few API calls. This data is far more accurate and timely than using discovery scans, manual entries into a CMDB, or relying on agents that might stop reporting on their state.

Second, you can use native, API-driven services to collect, filter, and analyze logs instead of maintaining and scaling the logging backend yourself. Pointing your logs at a bucket in an object store, or directing events to a streaming endpoint, means that you can spend less time on capacity planning and on ensuring the availability of the backend logging and search architecture.

In AWS, a best practice is to enable AWS CloudTrail and other service-specific logging to capture API activity globally and centralize the data for storage and analysis. You can direct AWS CloudTrail logs to Amazon CloudWatch Logs or other endpoints, so you can obtain events in a consistent format across compute, storage, and applications.

For server- and application-based logging that does not originate from the services themselves, you will still use traditional methods involving agents to collect and route events (syslog, third-party solutions, native operating system logging), but executing this approach with success is easier in AWS. Using services and features such as AWS CloudFormation, AWS OpsWorks, or Amazon Elastic Compute Cloud (EC2) user data, systems administrators can ensure that instances always have agents installed.

Equal in importance with collecting and aggregating logs is extracting meaningful information from the great volumes of log and event data generated by modern, complex architectures. Architects should consider detective controls end-to-end. You should not simply generate and store logs. Your information security function needs robust analytics and retrieval capabilities so that you can have insight into security-related activity. AWS provides solutions for big data workloads that are well suited to the task of parsing and analyzing security data.

Key AWS Services

The key AWS service that supports capturing key activities is AWS CloudTrail, which provides rich detail about API calls made in your AWS account. The following services and features are also important:

- **AWS Config** provides you with an AWS resource inventory, configuration history, and configuration change notifications to enable security and governance.

- **Amazon Elasticsearch Service** manages and scales a cluster of the popular open-source search and analytics engine Elasticsearch. You can use this solution to index, search, and render security data.
- **AWS CloudWatch Logs** allows you centralize logs into streams, natively integrating with features and services like VPC Flow Logs and AWS CloudTrail. CloudWatch Logs scales to ingest logs without the need to manage traditional infrastructure.
- **Amazon EMR** lets you write applications to parse and analyze logs at scale, while obviating the need to manage Hadoop and other data analysis clusters.
- **Amazon Simple Storage Service (S3) and Amazon Glacier** can be used to centralize storage and long-term archiving of log data.

Integrate Auditing Controls with Notification and Workflow

Security operations teams rely on the collection of logs and the use of search tools to discover potential events of interest, which may indicate unauthorized activity or unintentional change. However, simply analyzing collected data and manually processing information is insufficient to keep up with the volume of information flowing from modern, complex architectures. Analysis and reporting alone don't facilitate the assignment of the right resources to work an event in a timely fashion. A best practice for building a mature security operations team is to deeply integrate the flow of security events and findings into a notification and workflow system such a ticketing system, a bug/issue system, or other security information and event management (SIEM) system. This takes the workflow out of email and static reports, allowing you to route, escalate, and manage events or findings. Many DevOps organizations are now integrating security alerts into their chat/ Internet Relay Chat (IRC) tools or other collaboration and developer productivity platforms.

This best practice applies not only to security events generated from log messages depicting user activity or network events, but from changes detected in the infrastructure itself. The ability to detect change, determine whether a change was appropriate, and then route this information to the correct

remediation workflow is essential in maintaining and validating a secure architecture.

In AWS, routing events of interest and information reflecting potentially unwanted changes into a proper workflow is done using Amazon CloudWatch Events. This service provides a scalable rules engine designed to broker both native AWS event formats (such as AWS CloudTrail events), as well as custom events you can generate yourself. You build rules that parse events, transform them if necessary, then route such events to targets like an AWS Lambda function, Amazon Simple Notification Service (SNS) notification, or other targets.

Detecting change and routing this information to the correct workflow can be accomplished using AWS Config rules. AWS Config detects changes to in-scope services and generates events that can be parsed using AWS Config rules for rollback, enforcement of compliance policy, and forwarding of information to systems, such as change management platforms and operational ticketing systems.

Reducing the number of security misconfigurations introduced into a production environment is critical, so the more quality control and reduction of defects you can perform in the build process, the better. Modern continuous integration and continuous deployment (CI/CD) pipelines should be designed to test for security issues whenever possible. Using Amazon Inspector, you can perform configuration assessments for known common vulnerabilities and exposures (CVEs), assess your instances against security benchmarks and fully automate the notification of defects. Amazon Inspector runs on production instances or in a build pipeline, and it notifies developers and engineers when findings are present. You can access findings programmatically and direct them to a backlog and bug tracking systems.

Key AWS Services

The key AWS service that supports integrating auditing controls into notification and workflow systems is Amazon CloudWatch Events, which allows you to route events to a powerful rules engine. Rules then examine incoming events, parse the incoming values, and properly route the event to any number of targets, such as email or mobile devices, ticketing queues, and issue management systems. The following services and features are also important:

- **AWS Config Rules** enables you to create rules that automatically check the configuration of AWS resources recorded by AWS Config.
- **Amazon CloudWatch** must be enabled to facilitate collection of events and routing with CloudWatch Events.
- **Amazon CloudWatch API and AWS SDKs** can be used to create custom events in your own applications and inject them into CloudWatch Events for rule-based processing and routing.
- **Amazon Inspector** offers a programmatic way to find security defects or misconfigurations in your operating systems and applications. Because you can use API calls to access both the processing of assessments and the results of your assessments, integration of the findings into workflow and notification systems is simple. DevOps teams can integrate Amazon Inspector into their CI/CD pipelines and use it to identify any pre-existing issues or when new issues are introduced.

Resources

Refer to the following resources to learn more about AWS best practices for integrating auditing controls with notification and workflow.

Video

- Amazon CloudWatch Events[14]
- Essentials: Introducing AWS Config Rules[15]
- Introducing Amazon Inspector[16]

Documentation

- Amazon CloudWatch Events[17]
- AWS Config Rules[18]
- AWS Config Rules Repository (open source on GitHub)[19]
- Amazon Inspector[20]

Infrastructure Protection

Infrastructure protection encompasses control methodologies, such as defense in depth, necessary to meet best practices and industry or regulatory obligations. Use of these methodologies is critical for successful ongoing operations in either the cloud or on-premises.

Infrastructure protection is a key part of an information security program. It ensures that systems and services within your system are protected against unintended and unauthorized access and potential vulnerabilities. For example, you'll define trust boundaries (e.g., network boundaries and packet filtering), system security configuration and maintenance (e.g., hardening and patching), operating system authentication and authorizations (e.g., users, keys, and access levels), and other appropriate policy-enforcement points (e.g., web application firewalls and/or API gateways).

In AWS, there are a number of approaches to infrastructure protection. The following sections describe how to use these approaches:

- Protecting network and host-level boundaries
- System security configuration and maintenance
- Enforcing service-level protection

Protecting Network and Host-Level Boundaries

The careful management of your network topology and design forms the foundation of how you provide isolation and boundaries for resources within your environment. Because resources placed within your environment inherit the security properties of the underlying network, it is critical to establish an appropriate network design for the workload that ensures that only the desired network paths and routing is allowed. This can be done by leveraging multiple layers of protection to provide redundancy for the controls and mitigate the impact of a single layer misconfiguration that could allow inappropriate access.

When defining the network topology, consider which components of the system need to be public, for example, a customer-facing load balancer. Additionally, when you design connectivity, consider whether you need connectivity between your data center and AWS over a private network. Apply appropriate configurations to your virtual private cloud (VPC), subnets, routing tables,

network access lists (NACLs), gateways, and security groups to achieve the network routing as well as host-level protection.

A VPC created using Amazon Virtual Private Cloud allows you to define your network topology that spans an AWS Region with a private IP address range you determine. Within a VPC, you can create subnets in an Availability Zone. Each subnet has an associated route table that defines routing rules for managing the paths that traffic within the subnet takes; you can define publicly routable subnet by having a route that goes to an Internet Gateway attached to the VPC; the absence of a route to the Internet Gateway prevents instances from being directly reachable. A subnet can also have a network access list attached to it (stateless firewall). You can configure the ACLs to narrow the scope of traffic allowed. For example, you could only allow the port for the database engine for a subnet hosting the databases.

When a host is launched within a VPC, it has its own security group (stateful firewall). This firewall is outside the operating system layer and can be used to define rules for allowed traffic. You can also define relationships between security groups. For example, instances within a database tier security group only accept traffic from instances within the application tier.

When implementing a VPC design, keep in mind some key considerations for the IP address range that you choose for the VPC. Try to use non-overlapping IP addresses with your other VPCs or your data center. When designing NACL rules, consider that it's a stateless firewall and therefore both outbound and inbound rules need to be defined to meet your needs.

Key AWS Services

The key AWS service that supports the protection of network and host level boundaries is Amazon Virtual Private Cloud (VPC). This service provides the ability to create your private virtual network on AWS. The following services and features are also important:

- **Amazon VPC Security Groups**: Provide a per-host stateful firewall allowing you to specific traffic rules and define relationships to other security groups
- **AWS Direct Connect:** Allows you the ability to establish your own direct connectivity from your data center to your VPC.

Resources

Refer to the following resources to learn more about AWS best practices for protecting network and host-level boundaries.

Video

- Amazon VPC Deep Dive[21]

Documentation

- Amazon VPC Documentation[22]
- Amazon VPC Network Access Control Lists[23]
- Security Groups for Your VPC[24]
- Recommended Network ACL Rules for Your VPC[25]

System Security Configuration and Maintenance

The careful management of the security configurations of the running systems within your environment forms the foundation of how you will maintain robust, secure, scalable systems. The security posture of your systems is a function of the controls that are available and your own controls such as operating-system-based firewalls, CVE and vulnerability scanners, virus scanners, and any tools to help verify and maintain the integrity of your operating systems. These controls also form another layer in your defense in depth strategy.

When you define the approach for securing your system, consider the level of access needed for your system and take a least-privilege approach (e.g., only open the ports needed for communication, ensure that the operating system is hardened and that unnecessary tools and/or permissive configurations are disabled). You should apply appropriate configurations to your operating systems, including strong authorization mechanisms.

You should automate deployments and remove operator access to reduce your surface area. This can be achieved using EC2 Run Command. You should assess your baseline security exposures and perform routine vulnerability assessments when updates or deployments are pushed. Common vulnerabilities and exposures (CVE) scanning tools such as Amazon Inspector can be used to

achieve this and centralize security findings for analysis and remediation. You will want to ensure that your OS and application configurations, such as firewall settings and anti-malware definitions, are correct and up-date. You can use EC2 State Manager to define and maintain consistent OS configurations. Use EC2 Inventory to collect and query configuration about your instances and software installed. Leverage automated patching tools such as EC2 Patch Manager to help you deploy operating system and software patches automatically across large groups of instances.

Key AWS Services

The key AWS features that support the protection of systems are Amazon VPC security groups per-instance firewalls. The following services and features are also important:

- **Amazon Inspector** can be used to identify vulnerabilities or deviations from best practices in your guest operating systems and applications.
- **EC2 Run Command** provides a simple way of automating common administrative tasks such remotely executing shell scripts or PowerShell commands with granular access control and visibility, installing software updates, or making changes to the configuration of the OS.
- **EC2 State Manager** helps you define and maintain consistent OS configurations such as firewall settings and anti-malware definitions.
- **EC2 Inventory** helps you collect and query configuration and inventory information about your instances and the software installed on them.
- **EC2 Patch Manager** helps you select and deploy operating system and software patches automatically across large groups of instances.

Resources

Refer to the following resources to learn more about AWS best practices for system security configuration and maintenance.

Documentation

- EC2 Systems Manager[26]
- EC2 Run Command[27]
- EC2 State Manager[28]
- EC2 Inventory[29]

- EC2 Patch Manager[30]
- Replacing a Bastion Host with Amazon EC2 Systems Manager[31]

Enforcing Service-Level Protection

The careful management of the security configurations of service endpoints forms the foundation of how you will maintain secure and authorized access to these endpoints. This level of security is required to ensure that users and/or automated systems have exactly the level of access needed to perform their tasks (least privilege).

You can protect AWS service endpoints by defining policies using IAM. IAM can help you define policies for access to services and operations. However, in some services, you can also define fine-grained controls to specific resources within the service. Additionally, some resources have their own resource level policies. For example, Amazon S3 has bucket policies to define access levels to objects and/or entire buckets, and AWS Key Management Service (KMS) has policies to define administrators and users of the keys in the key management service. Use IAM and the resource policies to define a robust protection scheme for resources.

When defining a service-level protection approach, ensure that you apply a least-privilege methodology and set service-level access policies accordingly.

Key AWS Services

The key AWS service that supports service-level protection is AWS Identity and Access Management (IAM), which lets you define specific policies for many AWS resources. The following services and features are also important:

- **AWS Key Management Service (KMS)** allows you to set policies on the individual key.
- **Amazon Simple Storage Service (S3)** allows you to set bucket policies for each Amazon S3 bucket.

Resources

Refer to the following resources to learn more about AWS best practices for enforcing service-level protection.

Documentation

- VPC Security Groups[32]
- Using Key Policies in AWS KMS [33]
- Using Bucket Policies and User Policies[34]
- Amazon Inspector[35]

Data Protection

Before architecting any system, foundational practices that influence security should be in place. For example, *data classification* provides a way to categorize organizational data based on levels of sensitivity, and *encryption* protects data by way of rendering it unintelligible to unauthorized access. These tools and techniques are important because they support objectives such as preventing financial loss or complying with regulatory obligations.

In AWS, there are a number of different approaches to consider when addressing data protection, the following section describes how to use these approaches:

- Data classification
- Encryption/tokenization
- Protecting data at rest
- Protecting data in transit
- Data backup/replication/recovery

Data Classification

Data classification provides a way to categorize organizational data based on levels of sensitivity. This includes understanding what data types are available, where the data is located, and access levels and protection of the data (e.g., through encryption or access control). By carefully managing an appropriate data classification system along with each tier's protection requirements, you are able to map the controls and level of access/protection appropriate to the data. For example, public-facing content is available for anyone to access, whereas critical content is encrypted and stored in a protected manner that requires a key for decrypting the content.

By using resource tags, IAM policies, AWS KMS, and AWS CloudHSM, you can define and implement your policies for data classification. For example, if you have Amazon S3 buckets that contain highly critical data or EC2 instances that process confidential data, they can be tagged with a "DataClassification=CRITICAL" tag. You can define levels of access to the encryption keys through key policies to ensure that appropriate services have access to the sensitive content.

Data classification can also be implemented using third-party and open-source solutions hosted on AWS. You also define levels of access to different resources using IAM policies, as well as application-level controls to define resource access granted to each actor.

When you consider a data classification methodology, balance usability versus access. Also consider the multiple levels of access and nuances for implementing a secure but still usable approach for each level. Always consider a defense-in-depth approach. For example, require users to strongly authenticate to an application, and in addition you can also ensure that users come from a trusted network path and need to have access to the decryption keys.

Key AWS Services

The key AWS feature that supports data classification is resource tagging, which provides an ability to apply custom-defined tags for resources. The following services and features are also important:

- **AWS Key Management Service (KMS)**: Allows you to define encryption keys and access policies to them.

Resources

Refer to the following resources to learn more about AWS best practices for data classification.

Documentation

- Tagging your Amazon EC2 Resources[36]
- Using Key Policies in AWS KMS[37]
- Using Bucket Policies and User Policies[38]

Encryption/Tokenization

Encryption and tokenization are two important but distinct data protection schemes. *Tokenization* is a process that allows you to define a token to represent an otherwise sensitive piece of information (e.g., a token to represent a customer's credit card number). A token must be meaningless on its own. *Encryption* is a way of transforming content in manner that makes it unreadable without a secret key necessary to decrypt the content back into plain text. Both tokenization and encryption can be used to secure and protect content as appropriate.

By carefully defining your tokenization approach, you can provide additional protection for your content, and you can ensure that you meet various compliance requirements. For example, the scope of a credit card processing system can be narrowed if you leverage a token instead of a credit card number. You can define your own tokenization scheme by creating a look-up table in an encrypted Amazon Relational Database Service (RDS) database or an Amazon DynamoDB database and issue tokens to your end applications.

By defining an encryption approach, you can provide protection for your content against unauthorized users and against unnecessary exposure to authorized users. AWS provides a key management service for managing encryption keys (AWS KMS). This service provides durable, secure, and redundant storage for your master keys. You can define your key aliases as well as key-level policies. The policies help you define key administrators as well as key users. For example, a secret management system can be the only system that has access to the master key that encrypts the secrets for storage.

Additionally, the AWS CloudHSM service helps you meet corporate, contractual, and regulatory compliance requirements for data security by using dedicated hardware security module (HSM) appliances within the AWS Cloud. With AWS CloudHSM, you exclusively control the encryption keys and cryptographic operations performed by the HSM.

When defining an encryption/tokenization approach, consider the data classification model you've defined and the levels of access needed for each content. Consider the compliance requirements and needs around the content and how to strictly enable that approach. Also carefully consider the differences

and different use cases for tokenization versus encryption. Consider the key policies and access levels that would be provided for the user.

Key AWS Services

The key AWS service that supports encryption is the AWS Key Management Service (AWS KMS), which provides an easy-to-use, secure, and redundant key management service. The following services and features are also important:

- **AWS CloudHSM**: Provide a hardware security module for managing your keys
- **Amazon DynamoDB**: Provides a way to implement a fast NoSQL database. This can be used to store encrypted content for your tokens.

Resources

Refer to the following resources to learn more about AWS best practices for encryption/tokenization.

Video

- Encryption and Key Management in AWS[39]

Documentation

- Protecting Data Using Encryption[40]
- AWS Key Management Service[41]
- AWS CloudHSM[42]
- AWS KMS Cryptographic Details Whitepaper[43]

Protecting Data at Rest

Data at rest represents any content that you persist for any duration. This includes block storage, object storage, databases, archives, and any other storage medium on which data is persisted. By protecting your data at rest, you protect against unauthorized access, and can also protect against accidental leakage of content because it's useless if encrypted using a sufficiently strong encryption key.

Multiple AWS services provide built-in integration with the AWS Key Management Service (KMS) to allow a fast check box encryption of your persistent storage. Amazon S3 allows you to encrypt content by selecting a KMS key on object upload. Amazon Elastic Block Store (EBS) allows you to choose a master key to encrypt a block storage volume. Amazon Relational Database Service (RDS) allows you to choose an encryption key for encrypting DB instance storage at rest (including backups).

You also have the option of implementing your own data-at-rest approach. For example, you can encrypt content locally and store encrypted content. Amazon S3 provides you the facility to upload an already encrypted object, and it also provides the ability for you to upload an object along with an encryption key that's used in-memory to encrypt an object. To retrieve the object, you must supply the same key.

When implementing a data-at-rest protection approach, consider the data classification model for your organization to ensure that the content's protection reflects your internal goals. Additionally, consider any compliance or regulatory requirements. Also, ensure that you have provisioned the correct level of access to the keys, storage mediums, and any compute resources that have access to the content.

Key AWS Services

The key AWS feature that protects data at rest is the AWS Key Management Service, which provides an easy-to-use, secure, redundant, key management service. The following services and features are also important:

- **Amazon S3**: Provides an object storage service that integrates with KMS and allows you to supply your own keys.
- **Amazon Elastic Block Store (EBS)**: Provides block storage integrated with AWS KMS, but you can also perform block-level encryption with your operating system tools or third-party solutions.
- **Amazon Glacier**: Provides a long-term data archival solution that encrypts content at rest.

Resources

Refer to the following resources to learn more about AWS best practices for protecting data at rest.

Video

- Encryption and Key Management in AWS[44]

Documentation

- Protecting Amazon S3 Data Using Encryption[45]
- Amazon EBS Encryption[46]

Protecting Data in Transit

Data in transit is any content that gets transmitted from one system to another. This includes communication between servers within your environment as well as communication between other services and your end users. By providing the appropriate level of protection for your data in transit, you protect the confidentiality of your end users' content. When protecting your data in transit, selecting protocols that implement Transport Layer Security (TLS) is a common standard approach.

AWS services provide HTTPS endpoints for communication, thus providing encryption in transit when communicating with the AWS APIs. You have full control over your computing resources to implement encryption in transit across your services. Additionally, the AWS Certificate Manager (ACM) service provides you the ability to manage and deploy certificates for your domains.

Additionally, you can leverage VPN connectivity into your VPC or across your VPCs to facilitate encryption of traffic.

When planning for an encryption-in-transit approach, consider your use cases and the balance between encryption and ease of use. Consider the use of VPN connectivity from your data center to AWS and look into HTTPS for application-to-application communication in a secure manner. Using Elastic Load Balancing, Amazon CloudFront, and Amazon Certificate Manager make it easy to generate, deploy, and manage certificates used for TLS encryption.

Key AWS Services

The key AWS service that protects data in transit is **AWS Certificate Manager**, which helps you generate certificates used for establishing encrypted

transport between systems. The following services and features are also important:

- **Elastic Load Balancing: Classic Elastic Load Balancers and Application Load Balancers** help deploy and manage load balancers using secure endpoints.
- **Amazon CloudFront:** Supports encrypted endpoints for your content distributions.

Resources

Refer to the following resources to learn more about AWS best practices for protecting data in transit.

Documentation

- AWS Certificate Manager[47]
- How to address PCI Requirements for Data Encryption in Transit[48]
- Create a Classic Load Balancer with an HTTPS Listener[49]

Data Backup/Replication/Recovery

By defining your data backup, replication, and recovery approach, you protect yourself against deletion or destruction of data. A sound approach for data backup and replication helps protect you in case of a disaster. Properly secured and protected primary and secondary data sources can ensure continued business operations.

AWS provides you with multiple features and capabilities around data backup and replication. Amazon S3 is designed for 11 9's of durability for objects stored in the service, and it allows you to create copies of the content that can be copied to other locations and accounts for additional protection. Amazon RDS performs backups of your DB instances and allows you to replicate those instances to other locations. Snapshots can be taken of Amazon EBS volumes and copied across regions. Additionally, you can automate tasks and scheduled jobs to perform backups of resources. Consider using an AWS Lambda function to issue backups at an interval.

Amazon Glacier is a secure, durable, and extremely low-cost cloud storage service for data archiving and long-term backup. Use this service for cost-

effective storage of backup copies. These copies can be retrieved when needed for recovery whether it's for testing, regulatory issues, or disaster scenarios.

When defining your backup/replication/recovery approach, ensure that you review the scenarios under which you want to be protected as well as the nuances of each. For example, Amazon RDS will replicate any accidental changes, so if you have the backup you can be protected from accidental errors in addition to malicious actions. Ensure that you have a process defined for the recovery of your content. You should plan for a game day scenario to ensure that your approach is effective in the event of a disaster. Moreover, consider storing backups in a different account with a different set of credentials in case there is a major compromise of the primary account.

Key AWS Services

The key AWS service that supports data backup, replication, and recovery is **Amazon S3**. The following services and features are also important for backup and replication of data:

- **Amazon S3 Cross-Region Replication** is an Amazon S3 bucket-level feature that enables automatic, asynchronous copying of objects across buckets in different AWS Regions.
- **Amazon S3 lifecycle polices and versioning** allow you to implement a backup strategy and meet retention requirements.
- **Amazon Elastic Block Store snapshot operations** let you back up your volumes attached to EC2 instances.

Resources

Refer to the following resources to learn more about AWS best practices for data backup, replication, and recovery.

Documentation

- Amazon S3 Cross-Region Replication[50]
- Amazon S3 Object Lifecycle Management[51]
- Amazon S3 Object Versioning[52]
- Amazon EBS Snapshots[53]

- Getting Started with Amazon Glacier[54]
- Using AWS Lambda with Scheduled Events[55]

Incident Response

Even with extremely mature preventive and detective controls, organizations should still put processes in place to respond to and mitigate the potential impact of security incidents. The architecture of your workload strongly affects the ability of your teams to operate effectively during an incident, to isolate or contain systems, and to restore operations to a known-good state. Putting in place the tools and access ahead of a security incident, then routinely practicing incident response will help you ensure that your architecture is updated to accommodate timely investigation and recovery.

In AWS, there are a number of different approaches to consider when addressing incident response, the Clean Room section describes how to use these approaches.

Clean Room

In every incident, maintaining situational awareness is one of the most important principles. By using tags to properly describe your AWS resources, incident responders can quickly determine the potential impact of an incident. For example, tagging instances and other assets with an owner or work queue in a ticketing system allows the team to engage the right people more quickly. By tagging systems with a data classification or a criticality attribute, the impact of an incident can be estimated more accurately.

During an incident, the right people need to have access to isolate and contain the incident, then perform forensics and identify the root cause quickly. In some cases, the incident response team is actively involved in remediation and recovery as well. Determining how to get access for the right people in the middle of an incident delays the time it takes to respond, and can introduce other security weaknesses if access is shared or not properly provisioned while under pressure. Determine the access your team members need ahead of time, then regularly confirm that the access is appropriate and in place—or easily triggered—when needed.

In AWS you can use the power of the APIs to automate many of the routine tasks that need to be performed during an incident and subsequent investigations. For example, you can isolate an instance by changing the Security Groups associated with an instance or taking it out of rotation in a load balancer. Architecting your workload using Auto Scaling potentially allows the node under investigation to be removed from the capacity without affecting the availability of your applications.

For capturing the disk image or "as-is" configuration of an operating system, teams can use Amazon EBS snapshots and the Amazon EC2 Amazon Machine Image (AMI) APIs to capture the data and state of systems under investigation. Storing snapshots and related incident artifacts in Amazon S3 or Amazon Glacier ensures that the data will be available and retained as appropriate.

During an incident, before the root cause has been identified and the incident has been contained, it can be difficult to conduct investigations in an untrusted environment. Unique to AWS, security practitioners can use AWS CloudFormation to quickly create a new, trusted environment in which to conduct deeper investigation. The CloudFormation template can pre-configure instances in an isolated environment that contains all the necessary tools forensic teams need to determine the cause of the incident. This cuts down on the time it takes to gather necessary tools, isolated systems under examination, and ensures that the team is operating in a clean room.

Key AWS Services

Several key AWS services and features are critical to mature incident response:

- **Amazon Identity and Access Management (IAM)** should be used to grant appropriate authorization to incident response teams.
- **AWS CloudFormation** can be used to create a trusted environment for conducting deeper investigations.
- **Amazon EC2 APIs** can be used to help isolate instances and mitigate the impact of a security incident.

Resources

Refer to the following resources to learn more about AWS best practices for incident response.

Videos

- AWS re:Invent 2015 (SEC308) Wrangling Security Events in the Cloud[56]
- AWS re:Invent 2014 (SEC404) Incident Response in the Cloud[57]

Documentation

- Security Incident Response and Forensics on AWS[58]

Conclusion

Security is an ongoing effort. When incidents occur they should be treated as opportunities to improve the security of the architecture. Having strong authentication and authorization controls, automating responses to security events, protecting infrastructure at multiple levels, and managing well-classified data with encryption provides defense-in-depth that every business should expect. This effort is easier thanks to the programmatic functions and AWS features and services discussed in this paper.

AWS strives to help you build and operate architectures that protect information, systems, and assets while delivering business value. To make your architectures truly secure, you should use the tools and techniques discussed in this paper.

Contributors

The following individuals and organizations contributed to this document:

- Philip Fitzsimons, Sr Manager Well-Architected, Amazon Web Services
- Bill Shinn, Principal Security Solutions Architect, Amazon Web Services
- Sam Elmalak, Solutions Architect, Amazon Web Services

Further Reading

For additional help, please consult the following sources:

- AWS Well-Architected Framework Whitepaper[59]

Document History

May 26, 2017. Updated System Security Configuration and Maintenance section to reflect new AWS services and features.

Notes

[1] https://aws.amazon.com/architecture/well-architected/

[2] https://aws.amazon.com/compliance/shared-responsibility-model/

[3] https://www.youtube.com/watch?v=_wiGpBQGCjU&feature=youtu.be

[4] https://www.youtube.com/watch?v=-XARG9W2bGc&feature=youtu.be

[5] http://docs.aws.amazon.com/general/latest/gr/root-vs-iam.html

[6] http://docs.aws.amazon.com/IAM/latest/UserGuide/id_root-user.html

[7] http://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_passwords_account-policy.html

[8] http://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2_instance-profiles.html

[9] http://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_temp.html

[10] http://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_manage.html

[11] http://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_delegate-permissions.html

[12] http://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_managed-vs-inline.html

[13] http://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_manage.html

[14] https://www.youtube.com/watch?v=v1c9eYT6EWM

[15] https://www.youtube.com/watch?v=SpXZtN0Pjzw

[16] https://www.youtube.com/watch?v=ddz0JmCTTsU

[17] http://docs.aws.amazon.com/AmazonCloudWatch/latest/events/WhatIsCloudWatchEvents.html

[18] http://docs.aws.amazon.com/config/latest/developerguide/evaluate-config.html

[19] https://github.com/awslabs/aws-config-rules

[20] https://aws.amazon.com/inspector/

[21] https://www.youtube.com/watch?v=HexrVfuIY1k

[22] https://aws.amazon.com/documentation/vpc/

[23] http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_ACLs.html

[24] http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_SecurityGroups.html

[25] http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Appendix_NACLs.html

[26] https://aws.amazon.com/ec2/systems-manager/

[27] https://aws.amazon.com/ec2/systems-manager/run-command/

[28] https://aws.amazon.com/ec2/systems-manager/state-manager/

[29] https://aws.amazon.com/ec2/systems-manager/inventory/

[30] https://aws.amazon.com/ec2/systems-manager/patch-manager/

[31] https://aws.amazon.com/blogs/mt/replacing-a-bastion-host-with-amazon-ec2-systems-manager/

[32] http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_SecurityGroups.html

[33] http://docs.aws.amazon.com/kms/latest/developerguide/key-policies.html

[34] http://docs.aws.amazon.com/AmazonS3/latest/dev/using-iam-policies.html

[35] https://aws.amazon.com/documentation/inspector/

[36] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Using_Tags.html

[37] http://docs.aws.amazon.com/kms/latest/developerguide/key-policies.html

[38] http://docs.aws.amazon.com/AmazonS3/latest/dev/using-iam-policies.html

[39] https://www.youtube.com/watch?v=uhXalpNzPU4

[40] http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html

[41] https://aws.amazon.com/kms/

[42] https://aws.amazon.com/cloudhsm/

[43] https://d0.awsstatic.com/whitepapers/KMS-Cryptographic-Details.pdf

[44] https://www.youtube.com/watch?v=uhXalpNzPU4

[45] http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html

[46] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html

[47] http://docs.aws.amazon.com/acm/latest/userguide/acm-overview.html

[48] https://aws.amazon.com/blogs/security/how-to-address-the-pci-dss-requirements-for-data-encryption-in-transit-using-amazon-vpc/

[49] http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-create-https-ssl-load-balancer.html

[50] http://docs.aws.amazon.com/AmazonS3/latest/dev/crr.html

[51] http://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

[52] http://docs.aws.amazon.com/AmazonS3/latest/dev/ObjectVersioning.html

[53] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSSnapshots.html

[54] http://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-getting-started.html

[55] http://docs.aws.amazon.com/lambda/latest/dg/with-scheduled-events.html

[56] https://www.youtube.com/watch?v=uc1Q0XCcCv4

[57] https://www.youtube.com/watch?v=nzSrRvADh6g

[58] https://www.slideshare.net/AmazonWebServices/security-incident-response-and-forensics-on-aws

[59] https://d0.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf