AWS Core Security Audit Checklist

Version: 1.0 **Author:** Ivan Piskunov **Date:** 2024

Document Preface

This document provides a structured, actionable checklist for conducting a high-level security audit of fundamental Amazon Web Services (AWS) components. It is designed for security auditors, engineers, and cloud architects to quickly assess the security posture of an AWS environment.

Purpose: To serve as a practical field guide for evaluating critical security controls across key AWS services, leveraging both the AWS Management Console and the AWS Command Line Interface (CLI).

Intended Audience: Cloud Security Auditors, DevOps Engineers, Security Engineers, CISOs, Cloud Architects.

Disclaimer: This document is an independent compilation of best practices and does not constitute official AWS guidance. Always refer to the latest official AWS documentation for the most current and complete information. The author is not responsible for the use or misuse of this checklist.

Table of Contents

1. Identity and Access Management (IAM) Audit

Objective: Ensure the principle of least privilege is enforced and access keys are properly managed.

1.1. Audit IAM Users and Credentials

Console Check:

1. Navigate to **IAM > Users**.
2. Review the list of all users. Check the **'Access key status'** column. Any active keys older than 90 days should be investigated for rotation.
3. Click on a user. Under the **'Security credentials'** tab, check for:
    - **Console access**: Should be enabled only if necessary. If enabled, **MFA** should be marked as 'Enabled'.
    - **Access keys**: Note the creation date and last used date.
4. Go to **IAM > Credential report**. Generate and download the report. This CSV file provides a comprehensive view of all users' password and access key status, including last usage dates .

CLI Check:

- List all IAM users:

```
aws iam list-users
```

- List access keys for a specific user (`User_Name`):

```
aws iam list-access-keys --user-name User_Name
```

- Get a credential report (generates and downloads):

```
aws iam generate-credential-report
aws iam get-credential-report --output text --query Content | base64 -d > credential-report.csv
```

1.2. Audit IAM Policies and Permissions

Console Check:

1. Navigate to **IAM > Policies**.
2. Filter for **'Customer managed'** policies. Review each policy. Avoid policies with overly permissive actions (`*`) and resources (`*`). Policies should follow the principle of least privilege.
3. Use **IAM > Policy Simulator** to test what permissions a specific user or role has.
4. Navigate to **IAM > User groups** and **IAM > Roles**. Review the policies attached to each group and role, ensuring they are necessary and minimal .

CLI Check:

- List all customer-managed policies:

```
aws iam list-policies --scope Local
```

- Get details of a specific policy (replace `PolicyARN`):

```
aws iam get-policy --policy-arn PolicyARN
```

- Get the effective policy for a user:

```
aws iam get-account-authorization-details --filter User='User_Name'
```

2. Virtual Private Cloud (VPC) Audit

Objective: Ensure network isolation and control flow is properly configured.

2.1. Audit VPC Flow Logs

Console Check:

1. Navigate to **VPC > Your VPCs**.
2. Select a VPC. In the details pane below, check the '**Flow logs**' tab.
3. Ensure flow logs exist and are active. They should be sent to Amazon CloudWatch Logs or S3 for monitoring traffic patterns and detecting anomalies.

CLI Check:

- Describe flow logs for a specific VPC (`vpc-id`):

```
aws ec2 describe-flow-logs --filter Name=resource-id,Values=vpc-id
```

2.2. Audit Security Groups & NACLs

Console Check:

1. Navigate to **VPC > Security Groups**.
2. Review each security group's '**Inbound rules**'. A common best practice is to avoid rules with overly permissive sources (e.g., `0.0.0.0/0` for SSH/RDP). Restrict access to specific IP ranges where possible.
3. Navigate to **VPC > Network ACLs**. Review rules for each NACL associated with your subnets. Ensure they provide an additional layer of security where needed.

CLI Check:

- Describe all security groups and their rules:

```
aws ec2 describe-security-groups
```

- Describe Network ACLs:

```
aws ec2 describe-network-acls
```

3. VPN Configuration Audit

Objective: Ensure Site-to-Site VPN connections are secure and available.

3.1. Audit Site-to-Site VPN Connections

Console Check:

1. Navigate to **VPC > Site-to-Site VPN Connections**.
2. Select a VPN connection. Check its '**State**'. It should be '**available**'.
3. Check the '**Tunnel details**' tab. Review the status of both tunnels. Ideally, at least one tunnel should have a status of '**UP**' for high availability. Review the configured pre-shared keys and IP addresses for correctness .

CLI Check:

- Describe all VPN connections:

```
aws ec2 describe-vpn-connections
```

- Describe VPN connections filtered by state (e.g., 'available'):

```
aws ec2 describe-vpn-connections --filters Name=state,Values=available
```

4. Data Encryption & Key Management Audit

Objective: Ensure data at rest and in transit is encrypted by default.

4.1. Audit EBS Encryption

Console Check:

1. Navigate to **EC2 > Volumes**.
2. Review the '**Encryption**' column for each volume. It should show 'Encrypted' for sensitive data.
3. You can also check default encryption settings in **EC2 > Settings > Default encryption**.

CLI Check:

- Describe volumes, filtering for unencrypted volumes:

```
aws ec2 describe-volumes --filters Name=encrypted,Values=false
```

4.2. Audit S3 Bucket Encryption

Console Check:

1. Navigate to **S3**.
2. Click on a bucket. Go to the '**Properties**' tab.

3. Scroll down to '**Default encryption**'. It should be enabled, preferably with **SSE-KMS** for additional audit trails via AWS Key Management Service (KMS).

CLI Check:

- Get the encryption configuration for a specific bucket (`Bucket_Name`):

```
aws s3api get-bucket-encryption --bucket Bucket_Name
```

4.3. Audit AWS Key Management Service (KMS)

Console Check:

1. Navigate to **KMS > Customer managed keys**.
2. Select a key. Review its configuration:
    - **Key policy**: Ensure it follows the principle of least privilege.
    - **Rotation settings**: For CMKs, automatic annual key rotation should be enabled if supported by the key type.

CLI Check:

- List KMS keys:

```
aws kms list-keys
```

- Get key rotation status for a specific key (`Key_ID`):

```
aws kms get-key-rotation-status --key-id Key_ID
```

5. Backup & Recovery Audit

Objective: Ensure critical data and systems have resilient backup mechanisms.

5.1. Audit Amazon RDS Backups

Console Check:

1. Navigate to **RDS > Databases**.
2. Select a database instance. Check the '**Maintenance & backups**' tab.
3. Verify that '**Automated backups**' are enabled and the backup retention period is set according to your policy (e.g., 7 days or more).

CLI Check:

- Describe DB instances and check their backup retention period:

```
aws rds describe-db-instances --query 'DBInstances[*].{DBInstanceIdentifier:DBInstanceIdentifier,BackupRetentionPeriod:BackupRetentionPeriod}'
```

5.2. Audit AWS Backup Plans

Console Check:

1. Navigate to **AWS Backup**.
2. Check '**Backup plans**'. Ensure a backup plan exists and is assigned to resources you intend to protect (e.g., EFS, DynamoDB, EC2).
3. Go to '**Protected resources**' to verify resources are actually being backed up.

CLI Check:

- List backup plans:

```
aws backup list-backup-plans
```

6. Monitoring & Logging Audit

Objective: Ensure visibility into API activity and resource configuration changes.

6.1. Audit AWS CloudTrail

Console Check:

1. Navigate to **CloudTrail**.
2. Check '**Trails**' in the left pane. Ensure at least one Trail exists and is configured to apply to all Regions ('**Multi-region trail**').
3. Select the trail. Ensure '**Log file validation**' is enabled to detect tampering. Verify that '**Management events**' and '**Data events**' (S3, Lambda) are logged as required.
4. The trail should be configured to deliver logs to an S3 bucket that is highly restricted .

CLI Check:

- Describe trails:

```
aws cloudtrail describe-trails
```

6.2. Audit Amazon GuardDuty

Console Check:

1. Navigate to **GuardDuty**.
2. On the main page, check if the service is enabled. The status should show '**Enabled**'.
3. Review '**Findings**' for any active security alerts that need remediation.

CLI Check:

- List detectors (usually one per region):

```
aws guardduty list-detectors
```

- Get the status of a detector (`DetectorId`):

```
aws guardduty get-detector --detector-id DetectorId
```

Document Version History:

- Version 1.0 (2025-08-30): Initial security audit checklist compiled by Ivan Piskunov.

Important Note: This checklist is a starting point. A comprehensive audit should delve deeper into each service, consider industry-specific compliance frameworks (like PCI DSS, HIPAA), and include application-level security testing.