

OpenSecOps Foundation Account Properties

Standard Operating Procedure



Table of Contents

| | |
|---|-----------|
| 1 Introduction | 5 |
| Scope of SOP | 5 |
| 2 Basic Configuration | 6 |
| System Access | 6 |
| Enabled AWS Regions | 6 |
| Accounts | 6 |
| Sole Ownership | 6 |
| One Service and One Environment per Account | 7 |
| Email List | 7 |
| Ticketing and SLAs | 7 |
| Personal Accounts | 7 |
| Organisational Units | 8 |
| Jira | 9 |
| Deployment | 10 |
| Containers | 10 |
| Platform Agnosticism | 10 |
| AWS Copilot | 10 |
| Elastic Container Registry (ECR) | 11 |
| Elastic Container Service (ECS) | 11 |
| Permissions | 12 |
| Escalation of Privileges | 12 |
| Boundary Permission Policies | 12 |
| AWS Copilot | 12 |
| Service Control Policies (SCPs) | 12 |
| Logs | 14 |
| Incidents & Alarms | 14 |
| IAM Users | 15 |
| Use of IAM Users | 15 |
| IAM Password Policy | 15 |
| IAM User Password Rotation | 15 |
| Unused IAM Users | 15 |
| VPC | 15 |
| Default VPC | 15 |
| VPC Flow Logs | 15 |
| Unused Elastic IPs | 15 |
| VPC Peering | 15 |
| EC2 | 16 |
| Default EBS Volume Encryption | 16 |
| Instances Stopped For More Than 30 Days | 16 |
| Security Groups | 16 |
| Default Security Groups | 16 |
| Security Group World Ingress to Port 22 or 3389 | 16 |

| | |
|--|-----------|
| S3 Buckets | 16 |
| World Readable and/or Writable S3 Buckets | 16 |
| SQL Databases | 16 |
| DynamoDB Tables | 17 |
| Point-In-Time Recovery | 17 |
| CloudTrail | 17 |
| Security Hub | 17 |
| AWS Config | 18 |
| Inspector | 18 |
| Detective | 18 |
| Compute Optimizer | 18 |
| 3 Security Controls to Follow | 19 |
| Severity Levels | 19 |
| CRITICAL | 19 |
| HIGH | 20 |
| MEDIUM | 20 |
| LOW | 20 |
| 4 Auto-Remediations | 21 |
| S3.2: S3 Bucket Publicly Readable | 21 |
| S3.3: S3 Bucket Publicly Writable | 21 |
| S3.10: S3 buckets with versioning enabled should have lifecycle policies configured | 21 |
| EC2.7: Default EBS Encryption Not Enabled | 22 |
| CIS1.3: IAM User Credentials not used for 90 days | 22 |
| RDS.2: Prohibit Public RDS Access | 22 |
| RDS.4: RDS cluster snapshots and database snapshots should be encrypted at rest | 22 |
| RDS.6: Enhanced monitoring should be configured for RDS DB instances | 22 |
| RDS.11: RDS instances should have automatic backups enabled | 22 |
| RDS.13: RDS automatic minor version upgrades should be enabled | 22 |
| RDS.17: RDS DB instances should be configured to copy tags to snapshots | 22 |
| CIS2.8: Enable Customer CMK Key Rotation | 22 |
| CIS2.9: Enable VPC Flow Logs | 22 |
| CIS4.1: Disable World Ingress to Port 22 | 22 |
| CIS4.2: Disable World Ingress to Port 3389 | 23 |
| EC2.2: VPC Default Security Group Should Not Allow Traffic | 23 |
| EC2.4: Terminate Instances Stopped For More Than 30 Days | 23 |
| EC2.15: EC2 subnets should not automatically assign public IP addresses | 23 |
| ECR.1: ECR private repositories should have image scanning configured | 23 |
| ECR.3: ECR repositories should have at least one lifecycle policy configured | 23 |
| ECS.2: ECS services should not have public IP addresses assigned to them automatically | 23 |
| ECS.12: ECS clusters should use Container Insights | 23 |
| PCI.EC2.3: Release Unused Elastic IPs Older Than 30 Days | 23 |
| ELB.1: Application Load Balancer should be configured to redirect all HTTP | |

| | |
|--|-----------|
| requests to HTTPS | 23 |
| ELB.4: Application Load Balancer should be configured to drop invalid HTTP headers | 23 |
| ELB.5: Application and Classic Load Balancers logging should be enabled | 24 |
| 5 GuardDuty Incidents | 25 |
| EC2 Incidents | 25 |
| IAM Incidents | 26 |
| S3 Incidents | 27 |
| Kubernetes Incidents | 27 |
| Appendix A: Creating Incidents | 29 |
| 1 By Using CloudWatch Alarm Naming Conventions | 29 |
| 1.1 Usage | 29 |
| 1.2 Severity Examples | 29 |
| 1.3 Infrastructural vs Application Incidents | 29 |
| 1.4 Infrastructural Domain Examples | 29 |
| 1.5 Severities | 30 |

Document Versions

| Version | Date | Changes | Author |
|---------|------------|---|----------------|
| 1.0 | 2024-02-14 | First version | Peter Bengtson |
| 1.1 | 2025-04-07 | Replaced "Delegat" with "OpenSecOps" throughout | Peter Bengtson |

1 Introduction

A newly created AWS account in the OpenSecOps Foundation system is far from a blank slate.

Firstly, it has been configured to form a part of an overarching organisation with numerous requirements in such areas as security, reporting, costing, log collection and aggregation, monitoring, issue handling, ticketing and many other things.

Secondly, many AWS services have been enabled and configured for the benefit of the users of the account, who can use these services as they see fit in their daily work.

Thirdly, to help keep the account conform to the company's security requirements, it is constantly monitored for security issues and deviations from best practices. Some of these can be automatically remediated as they are encountered, ranging from closing overly permissive security group rules to snapshotting and terminating misbehaving server instances. Issues that cannot be remediated automatically are instead OpenSecOpsed to the appropriate team to fix themselves, via tickets in Jira with accompanying SLAs.

For these reasons, AWS accounts in the OpenSecOps Foundation organisation are opinionated. They have certain systemic properties, behaviours and requirements of which the users must be aware.

This SOP describes those properties, behaviours, and requirements.

Scope of SOP

This document is intended to give a high-level outline of what is available in a new account and of how the system is set up. For deeper information outside of this scope, cf the documents pertaining to the account level in OpenSecOps Foundation:

- OpenSecOps SOAR TDS
- OpenSecOps SOAR SOP
- OpenSecOps Foundation KMS Keys SOP
- OpenSecOps Foundation S3 Buckets SOP
- OpenSecOps Foundation DynamoDB SOP

Or, for the OpenSecOps Foundation system as a whole:

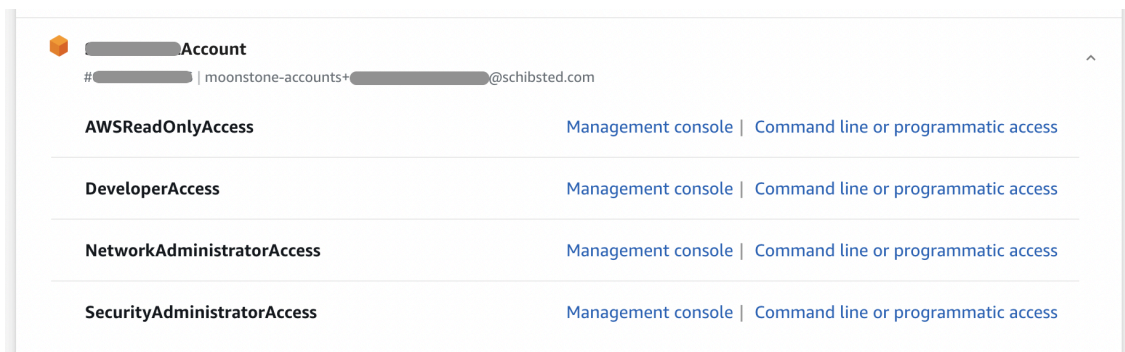
- OpenSecOps Foundation TDS
- OpenSecOps Foundation SOP


2 Basic Configuration

System Access

You will receive an email from the OpenSecOps Foundation system as part of your onboarding. The email will tell you how to log in using Single-Sign-On (SSO). When you log in the first time, you will be prompted to register one or more Multi-Factor Authentication (MFA) devices which then must be used every time you authenticate.

You will be presented with a list of accounts to which you have access. For each account, the access modes at your disposal will be listed. For example:



| | |
|---|--|
|  Account | |
| # Account moonstone-accounts+ Account @schibsted.com | |
| AWSReadOnlyAccess | Management console Command line or programmatic access |
| DeveloperAccess | Management console Command line or programmatic access |
| NetworkAdministratorAccess | Management console Command line or programmatic access |
| SecurityAdministratorAccess | Management console Command line or programmatic access |

Just choose the role you want to use and click “Management console” to go to the AWS console, or click “Command line or programmatic access” to obtain temporary credentials for command-line purposes.

Always use an access mode suitable for the type of work you are to do in the account. This is an important security consideration. Do not always use the most powerful access mode available to you. If just taking a look, use read-only access.

For more information about permissions, see the Permissions section.

Enabled AWS Regions

In the OpenSecOps Foundation system, all application work is restricted to the Stockholm AWS region (`eu-north-1`).

We also support North Virginia (`us-east-1`), but this region is intended only for system-level work and cannot be used by developers for application infrastructure, code, or data.

All other regions are disabled and can’t be accessed, not even by administrators.

Accounts

Sole Ownership

Each account in the OpenSecOps Foundation system is owned by exactly one team, or in the case of personal sandbox accounts, by exactly one person. There must be

no shared ownership of infrastructure within an account, nor must there be more than one team deploying to the same account.

The team or person owning the account is the sole contact point for the account, technical or otherwise. This is important for incident response, for enforcing SLAs to do with detected security issues, in attack and DR situations, etc.

One Service and One Environment per Account

Each account must host only one service and one environment (e.g., dev, test, prod). Security and compliance rules, requirements, controls, and auto-remediations vary with the type of environment.

Services may not share accounts even if they are managed by the same team. A service may sometimes need to be reassigned to another team: this should never require migration.

This means that a team uses *at least* three accounts per service: one account for managing the product's development cycle, another account for testing it, and another account for running the product in production. Under no circumstances must an account be used for multiple products or environments.

Email List

Each team must have a designated email distribution list to which automation can send email. Such email must reach all team members. There must be a process in place for each team to examine all email and to keep the distribution list up to date at all times.

Ticketing and SLAs

Each team must also have a process for dealing with tickets in Jira. Again, there must be a process to involve the whole team, and the process must take the urgency of the issues into consideration. Some issues must be dealt with immediately, others can wait a few days.

Infrastructural fixes must be allowed, when their severity level demands it, to take full priority over application feature development. The team manager must be prepared to allocate time accordingly and at times to stop feature development *for the whole team* until critical issues have been fixed.

Personal Accounts

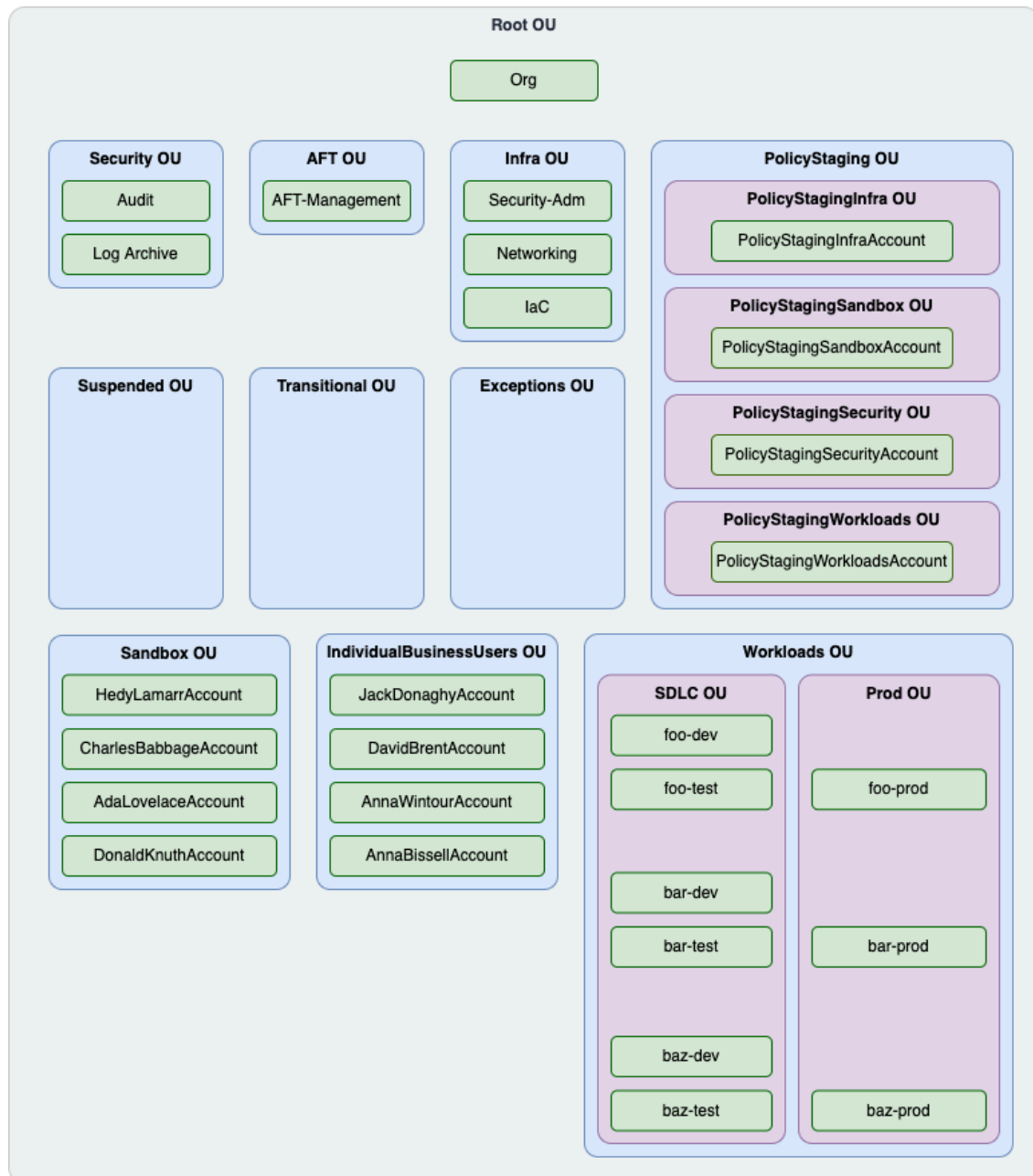
Every engineer has a personal sandbox account for their personal use in which experiments and tests can be carried out. These accounts are named after the individual (e.g. DiamandaGalasAccount, ScottyBowersAccount, KatyaZamolodchikovaAccount) and have a monthly spend cap and budget. Notifications will be sent when the budgeted max spend is approached or projected to be exceeded. The personal sandbox accounts may contain anything, including VPCs, but any VPC created cannot be reached from the internet.

Non-technical users such as business administrators also have personal sandbox accounts from which they can access the services and business tools they need.

All personal accounts whether technical or non-technical are subject to the same requirements as all other accounts: they are monitored for infrastructural issues, must have contact information valid at all times, the owner must be reachable via email and Jira tickets, and they are subject to escalating SLAs.

Organisational Units

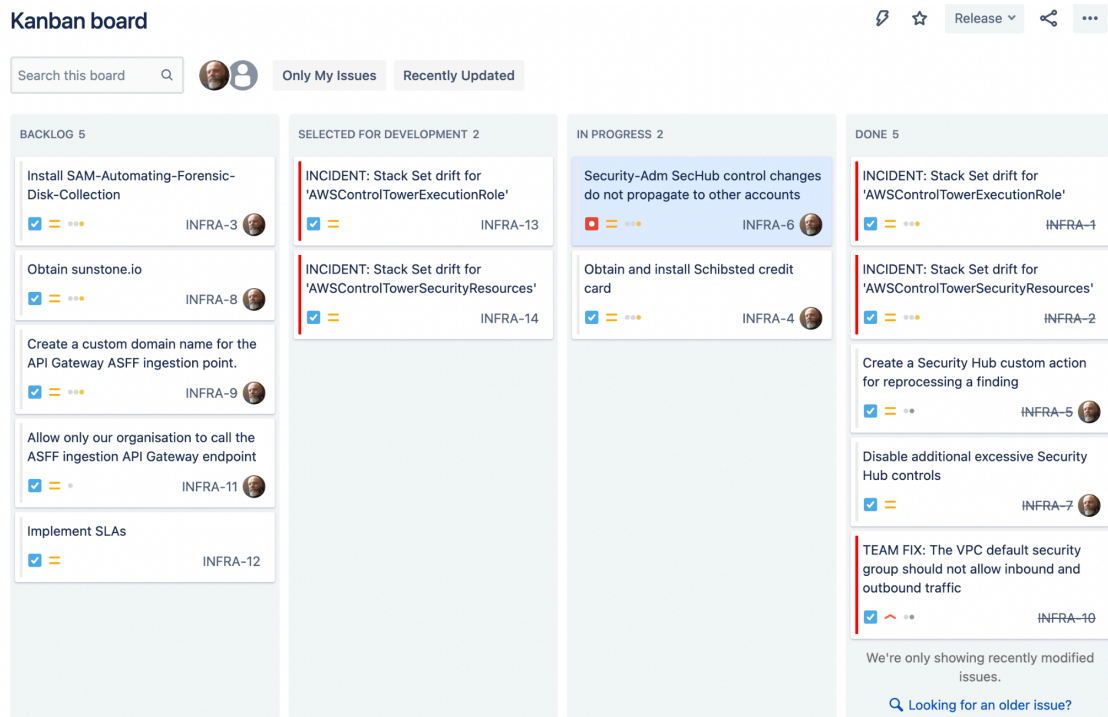
Below is a diagram of the organisational units in OpenSecOps Foundation. Note that OUs are not intended to mirror your company's organisation in any way. The OUs are purely technical in nature: they allow us to attach Service Control Policies (SCPs) appropriate for the security posture of each OU.



Jira

The OpenSecOps Foundation system is fully integrated with Jira Cloud. Jira boards are used by the teams for structuring and managing their work, as well as by automation to inject incidents and remediation tasks into the team's workflow. All work is thus visible in the same place.

Jira boards are assigned per account. Boards can be shared by more than one account, making it possible for teams to set up service-wide Jira boards (the default). Below is an example:



- **Incidents** ("INCIDENT:") – on both the infrastructural level and the application level – will automatically appear on the correct Jira board, for the correct team to attend to. The team drags the ticket to "In Progress" when they start working on it. When the incident has been dealt with, the teams can simply drag it to "Done".
- **Tasks** ("TEAM FIX:") assigned to the teams by OpenSecOps SOAR (Security Orchestration, Automation, and Response) will also appear here. Unlike incident tickets, teams cannot close these tickets by dragging them to "Done"; instead, OpenSecOps SOAR will close them and move them to "Done" when it detects the issue has been resolved.
- **Auto-remediations** ("AUTOFIXED:") performed by OpenSecOps SOAR will also appear here. Any modifications OpenSecOps SOAR does will also generate a ticket, the rationale being that actions should be taken to prevent the same auto-remediation from having to be done repeatedly. It is thus a prompt for your team and development process.

Tickets created by automation have a red bar to the left and will first appear in the "Selected for Development" column (not in the "Backlog" column, as it is expected that teams attend to them with priority). Cf the picture above for examples.

SOAR-created tickets for `TEAM FIXES` also have SLAs with escalation times depending on their severity. If a `TEAM FIX` isn't completed within the time allotted, it will automatically be escalated.

Deployment

Containers

The deliverable on the application level in the OpenSecOps Foundation system is the Docker container. For this reason, there are various tools and auto-remediations available to streamline the process of working with containers and to ensure their secure handling and storage.

Platform Agnosticism

To ensure the feasibility of retargeting to another cloud provider without application redesign, Docker containers in OpenSecOps Foundation:

- may not assume anything about the system that orchestrates them.
- may not assume they are being orchestrated by a particular orchestrator such as ECS, EKS, GKE, Cloud Run, Docker Swarm, etc.
- must always interface to the world in terms of only two things:
 - ports, and,
 - environment variables.
- may directly interface to platform-independent technology (such as SQL databases, Redis, Kafka, etc) using environment variables and ports.
- are welcome to use platform-specific services (such as AWS DynamoDB) but must use a library interface to abstract away the specifics. This allows other platform-specific technologies to be substituted in a platform-agnostic manner.

The above also means that applications can always and should always be tested and run locally during development. The platform independence of containers must be maintained at all times and in all stages of development.

AWS Copilot

AWS Copilot is used to manage containerised applications and services in OpenSecOps Foundation. Using Copilot, teams can set up a multi-account, multi-environment project in minutes, including pipelines and deployment to ECS Fargate.

Copilot also allows teams to write their own IAM policies and IAM roles and to configure their own security groups, all in a least-privilege environment and without any risk of escalation of privileges. The structure and configuration of pipelines are also under the control of the teams.

AWS Copilot also allows the team to monitor and manage the services post-deployment.

More information about AWS Copilot can be found here:

- <https://aws.github.io/copilot-cli>

Elastic Container Registry (ECR)

The counterpart to Docker Registry on AWS is ECR, the Elastic Container Registry. Docker images are uploaded to ECR for storage and to be automatically scanned for security vulnerabilities. It is from ECR that ECS will fetch the images to deploy.

The following automations are in place for ECR registries:

- ECR registries not configured to scan for security flaws will be set by an auto-remediation to scan uploaded images using enhanced scanning. The scanning is done once for each image, at the time it is uploaded. The results can be examined in the console; they can also be used as the basis of alarms and further automation.
- ECR registries without lifecycle configuration will be configured by an auto-remediation to keep only the two last uploaded images.
- ECR registries on OpenSecOps Foundation do not require ECR image tags to be set immutable as this might interfere with workflows using the 'latest' tag. Should those workflows not be used, there is an auto-remediation available to enforce immutable ECR tags. However, it is currently disabled.

The above applies to all ECR registries and their images. Copilot automatically creates and maintains ECR registries and Docker images for applications, but the above applies no matter how they are created and/or populated.

Elastic Container Service (ECS)

The orchestrator for Docker containers is AWS Elastic Container Service, also known as ECS. Separate ECS clusters are used for each service. We favour Fargate as the container deployment mode, making operation fully serverless.

The following automations are in place for ECS clusters:

- ECS services must not be assigned public IPs automatically by the cluster. If this feature is enabled, automation will disable it.
- If an ECS cluster doesn't have Container Insights enabled, auto-remediation will turn it on. Results collected by Container Insights can be seen in the console, and it is also possible to build CloudWatch alarms and further automation on them.

Permissions

Escalation of Privileges

Builders, whether part of an application development team or the administrators of the Foundation Team, have very wide permissions. These include the rights to create and manage VPCs, container clusters, DNS routing and much more.

Builders also have the right to create IAM Roles and IAM Policies. There is no security team or policy vending machine they need to interface to in order to obtain secure roles and policies. It's all been OpenSecOpsed to the Builders themselves.

However, empowering Builders to manage permissions also introduces the risk of Escalation of Privileges: it would be possible to write code that has wider privileges than its author. This is a well-known and easily exploitable attack vector.

Boundary Permission Policies

For this reason, Builders must specify a so-called pre-supplied Boundary Permission Policy for IAM Roles they create. Without it, IAM Roles simply cannot be created.

The Boundary Permission Policy is written and managed by the Foundation Team and will ensure that the Role you create is restricted to the permissions in the Boundary Permission policy, no matter what any policies attached or created by the Builder say. In this way, escalation of privileges cannot occur.

Boundary Permission Policies can't be detached, deleted or modified by Builders.

The Boundary Permission policy to use varies with the type of access used:

| Access mode | Boundary Permission Policy |
|-----------------------------|---|
| DeveloperAccess | developer-permission-boundary-policy |
| NetworkAdministratorAccess | network-administrator-permission-boundary-policy |
| SecurityAdministratorAccess | security-administrator-permission-boundary-policy |

From the above table it can be seen that Foundation Team administrators are not exempt from using Boundary Permissions. The code written by administrators is no less in need of protection than application code - quite the contrary, as the higher basic permissions of an administrator yield an even more attractive attack vector.

AWS Copilot

AWS Copilot supports Boundary Permissions. It can be configured to always attach the requisite boundary permission policy to any role it creates for your application, automating the whole process.

Service Control Policies (SCPs)

It should also be noted that apart from the Boundary Permission Policies there are Service Control Policies that further restrict the permissions of everyone using the system. The actual set of permissions granted through an IAM Policy you write will always be the logical intersection of your policy, the Boundary Permission used, and the system SCPs.

The SCPs protect the foundations of the system and enforce the basic properties and requirements of the system.

Logs

System logs from AWS Control Tower, including **CloudTrail** and **AWS Config** logs for all accounts and regions, are automatically collected and deposited in S3 buckets in the system-wide account for logging, `Log Archive`. This also applies to **GuardDuty** and **Security Hub** findings.

All local **CloudWatch Log groups** in an account are automatically streamed to the `Log Archive` account, where they are processed by log analysis software. This includes **VPC Flow Logs** and all logs from services that can log to CloudWatch.

This is also the way **application logging** is done: just make sure your application logs to CloudWatch and its logs will automatically be picked up, no matter their type. ECS/Fargate logs to CloudWatch Logs by default.

The retention time of CloudWatch log groups lacking an explicit setting will be set to 14 days by automation.

In addition to CloudWatch logs, any S3 bucket created locally to contain **Elastic Load Balancer** (ALB/ELB/NLB) or **CloudFront logs** will be automatically detected and aggregated to the `Log Archive` account. The bucket owner will also receive a Jira INFORMATIONAL incident ticket to inform them that logs now are being streamed from their bucket, and suggested further steps they might want to take in the local account.

There is an auto-remediation active for Elastic Load Balancers that creates an S3 bucket and sets up access logging for the load balancer to that bucket, if missing. This means that in OpenSecOps Foundation, all load balancer access logs are guaranteed to be aggregated without any user intervention whatsoever.

All logs in the `Log Archive` account are versioned, encrypted, tamper-proof, have access logging enabled, are sensibly life-cycled and saved for a total duration of 10 years. There is also additional automation to concatenate log files to a size suitable for true low-cost long-term Glacier deep archival storage.

Incidents & Alarms

In a system using OpenSecOps SOAR, CloudWatch Alarms can be used to create incidents on both the infrastructural level and the application level simply by naming them appropriately, such as `INFRA-FooBar-Alarm-CRITICAL`. This unifies incident handling across the entire system and all its applications.

It is important, indeed imperative, that all incident handling in the OpenSecOps Foundation system, be it infrastructural or application-related, goes through this mechanism as incidents created that way will automatically be integrated with the chosen ticketing system (Jira or ServiceNow). To ensure the long-term scalability and manageability of the system, there must not be multiple incident integration points. For more information, see OpenSecOps SOAR SOP or Appendix A of this manual.

IAM Users

Use of IAM Users

The use of IAM Users is strictly prohibited. All system access is federated using AWS SSO and temporary credentials. This includes service users, that is, IAM Users intended for automation.

In the very rare case that an IAM User is required, for instance when third-party software doesn't support federation, the Foundation Team must be contacted beforehand. IAM Users cannot be created except by infrastructural administrators.

You do *not* need an IAM User for CLI-based tools such as CloudFormation, Terraform, SAM or the CDK. Instead, use temporary, federated credentials. There are several ways of obtaining temporary credentials, of which the easiest is from the SSO login and account selection menu.

IAM Password Policy

For the rare cases when an IAM User must be used, the IAM User password policy (not to be confused with the SSO password policy) requires a password of 14 characters which must include uppercase, lowercase, digits and special characters.

IAM User Password Rotation

IAM User passwords must be rotated at least every 90 days. After this period, unchanged IAM credentials will be removed by automation. The last 24 passwords are remembered, preventing password reuse.

If you request an IAM User, you are responsible for building automation to perform password rotation automatically.

Unused IAM Users

IAM Users who haven't used their credentials for 90 days will have their credentials and access codes automatically revoked. The IAM User, though, will remain.

VPC

Default VPC

The default VPCs, which by default exist in each region of an account, have all been deleted. They behave slightly differently than other VPCs. For this reason, it is considered best practice to avoid using them at all.

VPC Flow Logs

VPCs created without Flow Logs will automatically have Flow Logs enabled through auto-remediation to dump REJECTED packages to a new CloudWatch Log group using a new, dedicated IAM Role and Policy.

Unused Elastic IPs

Unattached Elastic IP addresses are a security risk. Any unattached EIP 30 days or older will be deleted automatically.

VPC Peering

VPCs are not peered in OpenSecOps Foundation. Instead, VPC Service Endpoints/Private Links are to be used for access to internal services, whether from AWS or in the infrastructure or application layer.

EC2

Default EBS Volume Encryption

EBS storage volumes created in our supported regions have encryption enabled by default. You can only launch instance types that support EBS encryption.

Instances Stopped For More Than 30 Days

Instances which are left in a stopped state in the system constitute a security risk. OpenSecOps SOAR terminates any instance left stopped for more than 30 days. If you need an instance to survive in a mostly stopped state, script it to be started and patched at regular intervals.

Security Groups

Default Security Groups

Standard VPCs have a default Security Group, the use of which should be avoided as it by default permits all ingress and egress traffic. Never use the default Security Group. Instead, create a more restrictive Security Group for your VPC and use it instead. Always delete all rules in the default Security Group.

This best practice is considered so important that there is an active auto-remediation rule which will remove the routing rules of any VPC default Security Group within a couple of minutes of its creation.

Security Group World Ingress to Port 22 or 3389

World ingress to port 22 or 3389 will be removed by automation within a few minutes. This also applies to cases where the port is included in a range (such as ports 0 to 65535).

S3 Buckets

S3 buckets are file servers. As such, you are responsible for protecting them from unauthorised access, just like any other server you create.

World Readable and/or Writable S3 Buckets

There is automation to close all buckets open to the world unless tagged appropriately. Please see the OpenSecOps Foundation S3 Bucket SOP or Section 5 of this document for detailed information about how to legitimately create file servers open to the world.

SQL Databases

SQL Database storage in the OpenSecOps Foundation system must at all times be provided as a service from RDS. Running self-hosted databases on discrete EC2 instances is prohibited. We also strongly favour serverless DB deployment.

In practical terms, this means the norm is AWS Aurora Serverless v2, which is globally available in fully standards-compliant MySQL and PostgreSQL versions. AWS Aurora Serverless v2 instantly scales up and down to any capacity without

interruption, eliminating the need for manual operational database management and cluster scaling.

- <https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-serverless-v2.html>

Multi-AZ clusters should be used in production, but in development and testing environments, single-AZ installations are preferred. The production environments are monitored for compliance in terms of multi-AZ configuration; tickets will automatically be issued when production databases are non-compliant.

Sharing databases between services is prohibited. Every service must always have a dedicated database in each environment. Sharing databases is an anti-pattern; auto-scaling serverless databases such as Aurora v2 make it viable to work in a fully granular fashion using many smaller, individually auto-scaled databases rather than “one DB to rule them all”.

Databases will be discovered by automation. Logging, if not already configured, will automatically be enabled for Aurora Postgres DBs (including its Serverless variants) and plain vanilla RDS Postgres DBs. All database logs will be visible to the developers in their own account and will also be automatically aggregated to the Log Archive account for long-term storage.

DynamoDB Tables

DynamoDB is a very fast key-value store which spans storage needs from the strictly local to global, multi-regional tables with replication and point-in-time recovery. DynamoDB is much faster and cheaper than any SQL database and provides constant access times in the order of single milliseconds, no matter the size of the table.

Point-In-Time Recovery

There is automation to enable PITR for all DynamoDB tables to allow them to be restored to any point during the last 35 days. This is useful in a Disaster Recovery situation.

Please see the DynamoDB Table SOP or Section 5 of this document for detailed information about how to disable PITR for specific tables.

CloudTrail

There is exactly one organisation-wide, multi-regional CloudTrail in the system. It collects CloudTrail events from all accounts and stores them in the Log Archive account for archival purposes. The CloudTrail event logs are secure, encrypted, tamper-proof, and CIS-compliant. There is no need for anyone to create further CloudTrails, and the capability to do so is restricted to Cloud Administrators only.

Security Hub

AWS Security Hub is used to aggregate all security data in the OpenSecOps Foundation system. This includes info and findings from AWS native security services such as GuardDuty, CloudTrail, VPC Flow Logs, IAM Access Analyzer, AWS Inspector, AWS Macie, and others, and also from external security software through

its many integrations. AWS Security Hub findings in the ASFF format also form the basis on which all security automation is created.

Security Hub findings cannot be manipulated by non-admins. This means that issues can't be eliminated except by fixing the underlying issue. As a consequence, users do not have to close Jira tickets manually - OpenSecOps SOAR will do it for them when it detects that the issue has been resolved properly.

AWS Config

AWS Config is enabled everywhere appropriate and is used heavily by Security Hub, GuardDuty and other security services.

If you need to add your own AWS Config rules to an account, you can do so. All failed rule evaluations are also visible in Security Hub where they can be viewed using a Custom Insight. It is also possible to base CloudWatch alarms on such findings.

Inspector

AWS Inspector has been enabled in all accounts. The results of the security scans are available in the AWS Inspector console as well as in the form of findings in Security Hub.

You can create any number of Inspector runs, for instance, to assess only certain instances or instances with specific tags.

Detective

Amazon Detective makes it easy to investigate, analyse, and quickly identify the root cause of potential security issues or suspicious activities. Amazon Detective automatically collects log data from your AWS resources and uses machine learning, statistical analysis, and graph theory to help you visualise and conduct faster and more efficient security investigations. It is available both separately and as a seamlessly integrated part of AWS Security Hub.

Compute Optimizer

Compute Optimizer identifies workload patterns and recommends optimal AWS resources. Compute Optimizer analyses the configuration and resource utilisation of your workload to identify dozens of defining characteristics, for example, if a workload is CPU-intensive, if it exhibits a daily pattern, or if a workload accesses local storage frequently. The service processes these characteristics and identifies the hardware resource required by the workload. Compute Optimizer infers how the workload would have performed on various hardware platforms (e.g. Amazon EC2 instance types) or using different configurations (e.g. Amazon EBS volume IOPS settings, and AWS Lambda function memory sizes) to offer recommendations.

3 Security Controls to Follow

The following Security Hub controls apply to your account, which should be compliant with them at all times. All infrastructure is constantly monitored by Security Hub and acted upon by OpenSecOps SOAR.

For a fully up-to-date list of controls and detailed explanations of each item, please go to Security Hub and click on the links on the Security Standards tab.

Severity Levels

The levels of severity for a control or an incident are as follows:

Label

The severity value of the finding. The allowed values are the following.

- **INFORMATIONAL** - No issue was found.
- **LOW** - The issue does not require action on its own.
- **MEDIUM** - The issue must be addressed but not urgently.
- **HIGH** - The issue must be addressed as a priority.
- **CRITICAL** - The issue must be remediated immediately to avoid it escalating.

All work should be done in order of severity, starting with CRITICAL issues, then HIGH, then MEDIUM, and finally LOW. All issues must be addressed and SLAs must be adhered to, but some issues must be addressed before others.

Note the difference in urgency between the various levels of severity and try to understand the reasoning behind them, and how this affects the prioritisation of your team's work in the account.

CRITICAL

The issue must be remediated *immediately* to avoid it escalating.

For example, an open S3 bucket is considered a CRITICAL severity finding. Because so many actors scan for open S3 buckets, data in exposed S3 buckets is likely to be discovered and accessed by others.

In general, resources that are publicly accessible are considered critical security issues. You must treat CRITICAL findings with the utmost urgency. Your response time must be of the order of *minutes*.

HIGH

The issue must be addressed as a near-term priority.

For example, if a default VPC security group is open to inbound and outbound traffic, it is considered high severity. It is somewhat easy for a threat actor to compromise a VPC using this method. It is also likely that the threat actor will be able to disrupt or exfiltrate resources once they are in the VPC.

HIGH severity findings are to be treated as a near-term priority. You must take immediate remediation steps. Your response time must be of the order of *hours*, on the same day.

MEDIUM

The issue must be addressed as a mid-term priority.

For example, lack of encryption for data in transit is considered a medium severity finding. It requires a sophisticated man-in-the-middle attack to take advantage of this weakness. In other words, it is somewhat difficult. It is likely that some data will be compromised if the threat scenario is successful.

It is recommended that you investigate the implicated resource at your earliest convenience. This is *not* at the end of your sprint or in the next one.

LOW

The issue does not require immediate action on its own.

For example, failure to collect forensics information is considered LOW severity. This control can help to prevent future compromises, but the absence of forensics does not lead directly to a compromise.

You do not need to take immediate action on low severity findings, but they can provide context when you correlate them with other issues, and they should be acted upon as part of your weekly work cycle.

4 Auto-Remediations

The following is a list of the infrastructural remediations of Security Hub security Controls performed automatically by OpenSecOps SOAR. Infrastructure can have a compliance of PASSED or FAILED; the Security Hub controls keep track of issues as they change in compliance status over time.

Any failing control not up for auto-remediation will instead result in tickets being issued to the team, with accompanying SLAs.

It is important that automatic remediations are not seen as a mere convenience. They are first-line guardrails against security flaws created by those who do not yet know how to configure infrastructure in a way conformant with enterprise requirements.

Thus, an AUTOFIX or TEAMFIX ticket is a sign that there is more to learn. Moreover, the email and ticket will always contain pointers to documentation and remediation tips. Thus, there is really no need to have to deal with the same kind of AUTOFIX or TEAMFIX ticket again, provided the team has a sound process for integrating automation email and tickets into their shared workflow so that all can learn from shared experience.

Also be aware that statistics are kept per team, squad, app, environment, etc. The points awarded vary by issue severity, and they are considerably higher in production.

S3.2: S3 Bucket Publicly Readable

S3 buckets set to be publicly readable will be automatically closed for public access unless the tag `"soar:s3:request-publicly-readable"` is present on the bucket. SOC will be notified of your request via an incident. You will be contacted about your reasons for wanting to create a world-readable file server.

Please note that adding the tag after the fact will not reopen it for world access. You must redeploy the bucket with the correct tag, or manually add the tag then re-enable world access again.

S3.3: S3 Bucket Publicly Writable

S3 buckets set to be publicly writable will be automatically closed for public access unless the tag `"soar:s3:request-publicly-writable"` is present on the bucket at creation time. SOC will be notified of your request via an incident. You will be contacted about your reasons for wanting to create a world-writable file server.

Please note that adding the tag after the fact will not reopen it for world access. You must redeploy the bucket with the correct tag, or manually add the tag then re-enable world access again.

S3.10: S3 buckets with versioning enabled should have lifecycle policies configured

A lifecycle configuration will be added to the bucket. Noncurrent versions will be deleted after a year, and incomplete uploads after a day.

EC2.7: Default EBS Encryption Not Enabled

Enables default encryption of EBS volumes at the account level if not enabled. This will result in all new EBS volumes being encrypted. It will not affect running EBS volumes or their snapshots; only newly provisioned EBS volumes are affected.

CIS1.3: IAM User Credentials not used for 90 days

IAM Users who haven't used their credentials for 90 days will have their credentials and access codes revoked. The IAM User itself will remain.

RDS.2: Prohibit Public RDS Access

RDS databases set to be publicly accessible will have public access disabled.

RDS.4: RDS cluster snapshots and database snapshots should be encrypted at rest

An unencrypted DB cluster or instance snapshot will be deleted if empty. Otherwise, an attempt to encrypt the snapshot will be made, if the DB engine supports the operation (Aurora does *not*). If not supported, a TEAMFIX ticket will be issued instead.

RDS.6: Enhanced monitoring should be configured for RDS DB instances

Enables enhanced monitoring for your RDS DB instances. Enhanced Monitoring provides real-time metrics of the operating system that your RDS DB instance runs on. The monitoring period is set to 60 seconds.

RDS.11: RDS instances should have automatic backups enabled

Ensures Amazon Relational Database Service instances have automated backups enabled and the backup retention period is greater than or equal to seven days.

RDS.13: RDS automatic minor version upgrades should be enabled

Ensures that the latest minor version updates to the relational database management system (RDBMS) always are installed. These upgrades might include security patches and bug fixes. Keeping up to date with patch installation is an important step in securing systems.

RDS.17: RDS DB instances should be configured to copy tags to snapshots

Snapshots should be tagged in the same way as their parent RDS database instances. This auto-remediation ensures that snapshots inherit the tags of their parent database instances.

CIS2.8: Enable Customer CMK Key Rotation

Customer CMK KMS Keys without yearly key rotation will have such rotation enabled automatically.

CIS2.9: Enable VPC Flow Logs

VPCs without Flow Logs will have Flow Logs enabled and dump REJECTEd packages to a new CloudWatch Log group using a new Role and Policy.

CIS4.1: Disable World Ingress to Port 22

World ingress to port 22 will be removed entirely. This also applies to cases where port 22 is included in a range (such as ports 0 to 65535) in which case the whole range will be removed.

CIS4.2: Disable World Ingress to Port 3389

World ingress to port 3389 will be removed entirely. This also applies to cases where port 3389 is included in a range (such as ports 0 to 65535) in which case the whole range will be removed.

EC2.2: VPC Default Security Group Should Not Allow Traffic

The default Security Group for a VPC is always created with full ingress and egress rights, which is reason to avoid using them at all. This remediation removes all ingress and egress rights for a VPC Default Security Group, provided the security group is untouched. If any changes have been done to the SG, it will be kept as is and a TEAMFIX ticket will be created instead.

EC2.4: Terminate Instances Stopped For More Than 30 Days

Instances that are left in a stopped state in the system are a security risk. This remediation terminates any instance left stopped for more than 30 days.

EC2.15: EC2 subnets should not automatically assign public IP addresses

If a VPC subnet is configured to automatically assign public IP addresses, then this setting is disabled.

ECR.1: ECR private repositories should have image scanning configured

If an ECR private repository doesn't have image scanning configured, this remediation will configure it to scan uploaded Docker images using enhanced scanning once when the image is uploaded.

ECR.3: ECR repositories should have at least one lifecycle policy configured

If an ECR repository doesn't have a lifecycle policy, this remediation will add one that keeps only the last uploaded Docker image.

ECS.2: ECS services should not have public IP addresses assigned to them automatically

If an ECS service assigns public IP addresses to its tasks, this will be disabled.

ECS.12: ECS clusters should use Container Insights

ECS clusters will have Container Insights activated if not enabled.

PCI.EC2.3: Release Unused Elastic IPs Older Than 30 Days

Unattached Elastic IP addresses older than a month are a security risk. This remediation will remove any EIP older than 30 days.

ELB.1: Application Load Balancer should be configured to redirect all HTTP requests to HTTPS

This autoremediation will set up HTTP to HTTPS redirection for public facing ALBs with a matching SSL certificate. If no such certificate is detected, a ticket will be created to the team instead.

ELB.4: Application Load Balancer should be configured to drop invalid HTTP headers

By default, Application Load Balancers are not configured to drop invalid HTTP header values. Removing these header values prevents HTTP desync attacks.

ELB.5: Application and Classic Load Balancers logging should be enabled

Creates an S3 bucket for load balancer access logs and sets up replication of its contents to the Log Archive account. This guarantees that load balancer logs always are collected and aggregated for compliance.

5 GuardDuty Incidents

Amazon GuardDuty is a threat detection service that continuously monitors for malicious activity and unauthorised behaviour. GuardDuty findings automatically appear in Security Hub.

Its findings do not have a state of PASSED or FAILED but instead represent incidents, i.e., threats happening at the current time with some degree of urgency; thus OpenSecOps SOAR will choose to act or not act upon them depending on their severity (CRITICAL, HIGH, MEDIUM, LOW, and INFORMATIONAL).

GuardDuty findings apply to four main infrastructural types:

- EC2 – possibly compromised server instances (Bitcoin mining, etc)
- S3 – possibly compromised S3 buckets
- IAM – possibly compromised user credentials (including federated users)
- EKS – possibly compromised Kubernetes pods

For EC2 instances, remediation involves terminating the instance, with prior snapshotting of attached volumes for later examination by a forensics team. An incident to SOC to investigate the circumstances is also created.

For S3 buckets, remediation involves modifying access.

For IAM users, remediation may involve credentials being revoked.

As S3 and IAM integrity already is handled by the Security Hub control automation, the first phase of GuardDuty automation is geared towards cases involving EC2 instances.

The full list of GuardDuty findings can be found here:

https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_finding-types-active.html.

EC2 Incidents

The page listing EC2 findings is here:

https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_finding-types-ec2.html

Here is a list of the EC2 incident findings we are supporting. We will auto-remediate all of them, provided they have a Severity of MEDIUM, HIGH or CRITICAL, which is most of them. For full details, please see the link above.

- Backdoor:EC2/C&CAActivity.B
- Backdoor:EC2/C&CAActivity.B!DNS
- Backdoor:EC2/DenialOfService.Dns
- Backdoor:EC2/DenialOfService.Tcp
- Backdoor:EC2/DenialOfService.Udp
- Backdoor:EC2/DenialOfService.UdpOnTcpPorts
- Backdoor:EC2/DenialOfService.UnusualProtocol
- Backdoor:EC2/Spambot
- Behavior:EC2/NetworkPortUnusual

- Behavior:EC2/TrafficVolumeUnusual
- CryptoCurrency:EC2/BitcoinTool.B
- CryptoCurrency:EC2/BitcoinTool.B!DNS
- Impact:EC2/AbusedDomainRequest.Reputation
- Impact:EC2/BitcoinDomainRequest.Reputation
- Impact:EC2/MaliciousDomainRequest.Reputation
- Impact:EC2/PortSweep
- Impact:EC2/SuspiciousDomainRequest.Reputation
- Impact:EC2/WinRMBruteForce
- Recon:EC2/PortProbeEMRUnprotectedPort
- Recon:EC2/PortProbeUnprotectedPort
- Recon:EC2/Portscan
- Trojan:EC2/BlackholeTraffic
- Trojan:EC2/BlackholeTraffic!DNS
- Trojan:EC2/DGADomainRequest.B
- Trojan:EC2/DGADomainRequest.C!DNS
- Trojan:EC2/DNSDataExfiltration
- Trojan:EC2/DriveBySourceTraffic!DNS
- Trojan:EC2/DropPoint
- Trojan:EC2/DropPoint!DNS
- Trojan:EC2/PhishingDomainRequest!DNS
- UnauthorizedAccess:EC2/MaliciousIPCaller.Custom
- UnauthorizedAccess:EC2/MetadataDNSRebind
- UnauthorizedAccess:EC2/RDPBruteForce
- UnauthorizedAccess:EC2/SSHBruteForce
- UnauthorizedAccess:EC2/TorClient
- UnauthorizedAccess:EC2/TorRelay

IAM Incidents

https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_finding-types-iam.html

IAM incidents are based on machine learning models. Incidents will be created for unusual logins and API behaviour patterns for individual users. Time, place, and the nature of the APIs used are taken into account.

- CredentialAccess:IAMUser/AnomalousBehavior
- DefenseEvasion:IAMUser/AnomalousBehavior
- Discovery:IAMUser/AnomalousBehavior
- Exfiltration:IAMUser/AnomalousBehavior
- Impact:IAMUser/AnomalousBehavior
- InitialAccess:IAMUser/AnomalousBehavior
- PenTest:IAMUser/KaliLinux
- PenTest:IAMUser/ParrotLinux
- PenTest:IAMUser/PentooLinux
- Persistence:IAMUser/AnomalousBehavior
- Policy:IAMUser/RootCredentialUsage
- PrivilegeEscalation:IAMUser/AnomalousBehavior
- Recon:IAMUser/MaliciousIPCaller
- Recon:IAMUser/MaliciousIPCaller.Custom
- Recon:IAMUser/TorIPCaller
- Stealth:IAMUser/CloudTrailLoggingDisabled
- Stealth:IAMUser/PasswordPolicyChange
- UnauthorizedAccess:IAMUser/ConsoleLoginSuccess.B
- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.InsideAWS

- UnauthorizedAccess:IAMUser/InstanceCredentialExfiltration.OutsideAWS
- UnauthorizedAccess:IAMUser/MaliciousIPCaller
- UnauthorizedAccess:IAMUser/MaliciousIPCaller.Custom
- UnauthorizedAccess:IAMUser/TorIPCaller

S3 Incidents

https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_finding-types-s3.html

- Discovery:S3/MaliciousIPCaller
- Discovery:S3/MaliciousIPCaller.Custom
- Discovery:S3/TorIPCaller
- Exfiltration:S3/MaliciousIPCaller
- Exfiltration:S3/ObjectRead.Unusual
- Impact:S3/MaliciousIPCaller
- PenTest:S3/KaliLinux
- PenTest:S3/ParrotLinux
- PenTest:S3/PentooLinux
- Policy:S3/AccountBlockPublicAccessDisabled
- Policy:S3/BucketAnonymousAccessGranted
- Policy:S3/BucketBlockPublicAccessDisabled
- Policy:S3/BucketPublicAccessGranted
- Stealth:S3/ServerAccessLoggingDisabled
- UnauthorizedAccess:S3/MaliciousIPCaller.Custom
- UnauthorizedAccess:S3/TorIPCaller

Kubernetes Incidents

https://docs.aws.amazon.com/guardduty/latest/ug/guardduty_finding-types-kubernetes.html

- CredentialAccess:Kubernetes/MaliciousIPCaller
- CredentialAccess:Kubernetes/MaliciousIPCaller.Custom
- CredentialAccess:Kubernetes/SuccessfulAnonymousAccess
- CredentialAccess:Kubernetes/TorIPCaller
- DefenseEvasion:Kubernetes/MaliciousIPCaller
- DefenseEvasion:Kubernetes/MaliciousIPCaller.Custom
- DefenseEvasion:Kubernetes/SuccessfulAnonymousAccess
- DefenseEvasion:Kubernetes/TorIPCaller
- Discovery:Kubernetes/MaliciousIPCaller
- Discovery:Kubernetes/MaliciousIPCaller.Custom
- Discovery:Kubernetes/SuccessfulAnonymousAccess
- Discovery:Kubernetes/TorIPCaller
- Execution:Kubernetes/ExecInKubeSystemPod
- Impact:Kubernetes/MaliciousIPCaller
- Impact:Kubernetes/MaliciousIPCaller.Custom
- Impact:Kubernetes/SuccessfulAnonymousAccess
- Impact:Kubernetes/TorIPCaller
- Persistence:Kubernetes/ContainerWithSensitiveMount
- Persistence:Kubernetes/MaliciousIPCaller
- Persistence:Kubernetes/MaliciousIPCaller.Custom
- Persistence:Kubernetes/SuccessfulAnonymousAccess

- Persistence:Kubernetes/TorIPCaller
- Policy:Kubernetes/AdminAccessToDefaultServiceAccount
- Policy:Kubernetes/AnonymousAccessGranted
- Policy:Kubernetes/ExposedDashboard
- Policy:Kubernetes/KubeflowDashboardExposed
- PrivilegeEscalation:Kubernetes/PrivilegedContainer

Appendix A: Creating Incidents

1 By Using CloudWatch Alarm Naming Conventions

This is the quickest and most lightweight way of creating incidents, provided the incident can be based on a CloudWatch alarm.

OpenSecOps SOAR monitors every CloudWatch alarm in all regions of all member accounts. By simply including certain strings in the CloudWatch alarm name, incidents will be created depending on that information.

1.1 Usage

To create an incident when a CloudWatch alarm is raised, simply include the severity level in capitals at the very beginning or end of its name, followed or preceded by a hyphen, respectively.

1.2 Severity Examples

A few example CloudWatch Alarm names that will create incidents when triggered:

- INFORMATIONAL-App-Alarms_e3540a15f0
- LOW-Alarm-with-ML
- Whatever-Alarm-You-Have-12345-MEDIUM
- HIGH-as-a-kite-2418
- Its-getting-CRITICAL

A few names that will not:

- informational-App-Alarms_e3540a15f0
- Low-Alarm-with-ML
- Whatever-MEDIUM-Alarm-You-Have-12345
- HIGHas-a-kite-2418
- ItsgettingCRITICAL

1.3 Infrastructural vs Application Incidents

The incident will by default be created for the application domain. To instead create an incident pertaining to the infrastructural domain, just include the string `INFRA` in capitals anywhere in the alarm name.

1.4 Infrastructural Domain Examples

A few names that will create an infrastructural incident:

- INFORMATIONAL-INFRA-App-Alarms_e3540a15f0
- LOW-INFRA-Alarm-with-ML
- Whatever-INFRACTIVE-Alarm-You-Have-12345-MEDIUM

A few names that won't:

- INFORMATIONAL-Infra-App-Alarms_e3540a15f0
- LOW-Infra-Alarm-with-ML
- Whatever-infractive-Alarm-You-Have-12345-MEDIUM

1.5 Severities

The standard severities to use are `LOW` and `MEDIUM`.

The two highest severities, `HIGH` and `CRITICAL`, are show-stoppers that require the team to address the issue with extreme priority. Use them with extreme discretion and consult the Foundation Team beforehand.