AWS Landing Zone

Prepared by AWS Professional Services
Last updated: December 22, 2022

Table of Content

Confidentiality of Contents

The contents of this document are shared only under NDA and should not be reattributed.

Amazon Web Services Disclaimer

This document is not legally-binding, and is not an offer to contract that can be accepted by either party. All responses, descriptions and observations in this document are informational and are provided solely for discussion purposes. Neither party will have any obligation or liability with respect to the matters described in this document. All obligations must be set forth in a separate definitive agreement executed by the parties addressing such matters, provided, however, that neither party will have any liability for any failure or refusal to enter into a definitive agreement for any reason. Amazon Web Services, Inc. (AWS) has provided responses based on its current knowledge, but these responses may change at any time due to a variety of factors, including without limitation, changes to your requirements, the capabilities of any third party you select to assist with implementation, and changes to AWS's service offerings. AWS does not make any representations or warranties of any kind in this document. Any use of the AWS service offerings will be governed by the AWS Customer Agreement available at http://aws.amazon.com/agreement/ (or other definitive written agreement between the parties), not this document. AWS does not accept any terms or conditions included in this document that conflict with or are in addition to the terms and conditions set forth in the AWS Customer Agreement.

Document Objective

This document serves as a High Level Design Document detailing the setup of the "Landing zone Architecture using AWS Control Tower", for Virgin Media O2 (VMO2 from here). This document assumes a familiarity with AWS terms and is written for a generally technical audience.

Background

This design proposal for "Landing zone Architecture using AWS Control Tower" is *subject to client review, changes and approvals before implementation*. This design is also subject to business and IT strategy review. By the end of the document, the reader will understand VMO2's vision for its AWS environment and the items that should be addressed in order to implement this strategy. There is an expectation that the strategy will continue to evolve as additional perspectives are evaluated.

Contributing Participants

The key participants from for the engagement:

Name	Email
Matthew Handley	matt.handley@virginmediao2.co.uk
Adrian Cleary	adrian.cleary@virginmediao2.co.uk
Ben Littlewood	Ben.Littlewood@virginmedia.co.uk

The key contributors from AWS for the engagement:

Name	Title	Email
Luis Lopez Fernandez	Cloud Infrastructure Architect	luislf@amazon.com

1 AWS Control Tower

In order to simplify the creation and governance of VMO2's Landing Zone, AWS Control Tower service will be leveraged. AWS Control Tower automates the setup of a new landing zone using best-practices blueprints for identity, federated access, and account structure. Following features come along with AWS Control Tower out of the box:

Account Factory

The account factory automates provisioning of new accounts as part of an AWS organization. As a configurable account template, it helps in standardizing the provisioning of new accounts with pre-approved account configurations. It allows in configuration of account factory with pre-approved network configuration and region selections. Self-service for builders to configure and provision new accounts using AWS Service Catalog is also possible.

Preventive & Detective Guardrails

Guardrails are pre-packaged governance rules for security, operations, and compliance that can be selected and applied AWS Organization wide or to specific groups of accounts. A guardrail is expressed in plain English, and enforces a specific governance policy for an AWS environment that can be enabled within an AWS Organizations organizational unit (OU). Each guardrail contains two dimensions: it can be either preventive or detective, and it can be either mandatory or optional. Preventive guardrails establish intent and prevent deployment of resources that don't conform to your policies (for example enable AWS CloudTrail in all accounts). Detective guardrails (disallow public read access for S3 buckets) continuously monitor deployed resources for nonconformance. Control Tower automatically translates guardrails into granular AWS policies by:

- Establishing a configuration baseline using AWS CloudFormation
- Preventing configuration changes of the underlying implementation using service control policies (for preventive guardrails)
- Continuously detecting configuration changes through AWS Config rules (for detective guardrails)
- Updating guardrail status on the Control Tower dashboard

Mandatory & Optional Guardrails

AWS Control Tower offers a curated set of guardrails based on AWS best practices and common customer policies for governance. Mandatory guardrails can be automatically leveraged as part of a landing zone setup. Some examples of mandatory guardrails include:

- Disallow changes to IAM roles set up for AWS Control Tower
- Disallow public read access to log archive
- Disallow policy changes to log archive

Strongly recommended guardrails can also be applied at any time on OUs. All accounts provisioned under enabled OUs will automatically inherit those guardrails. Some examples of strongly recommended guardrails include:

- Disallow public write access to Amazon Simple Storage Service (Amazon S3) buckets
- Disallow access as a root user without multi-factor authentication
- Enable encryption for Amazon Elastic Block Store (Amazon EBS) volumes attached to Amazon Elastic Compute Cloud (Amazon EC2) instances

Dashboard

The Control Tower dashboard gives continuous visibility into an AWS environment. Viewing the number of OUs and accounts provisioned, the number of guardrails enabled, and the checking the status of OUs and accounts against those guardrails is possible. A list of noncompliant resources with respect to enabled guardrails can also be seen.

2 AWS Account Structure

Organizational Units (OU's) represent policy enforcement points for Service Control Policies (SCP's), AWS Config rules, Tagging Policies and resource sharing. All these controls are typically applied by the platform teams and the account users will not have permission to change organizational policies.

The AWS Account Structure within AWS Organizations is a fundamental building block in the overall landing zone and they have key design objectives identified as:

- *Balance developer agility and platform management/governance.*
- *Have the capabilities to test and apply different controls/policies to production and non-production environments.*
- *A solution should be in place to allow the testing and evolution of policies and controls which does not impact non-production/production as VMO2 evolve their platform.*
- *A simplify overall OU structure should be used that supports easier automation, reporting and auditing. To achieve that, minimize the depth of the overall hierarchy and where the polices are applied.*
- *This structure should be able to support up and increasing number of accounts.*

In order to meet these goals, the following technical decisions have been made that influence the design shown below:

1. *Initial Account structure will be based on business organizations (CIO, CTO, CDO), then divided based on purpose; Production and Non-Production where Non-Production is category that covers Dev/Test accounts/workloads.*
2. *Production and Non-Production OUs should be placed under each level 1 organization to create a logical account grouping with duties separation.*
3. *A separate Organization should be created for development and testing of new policies.*
4. *Alignment between OU controls (SCP's, Tag Policies, resource sharing, and config rules) and account controls (IAM and Network) is required to provide the required separation and permission boundaries.*

The diagram below illustrates the proposed structure which is orientated fewer OU control points rather than OUs for teams, functions and/or applications.

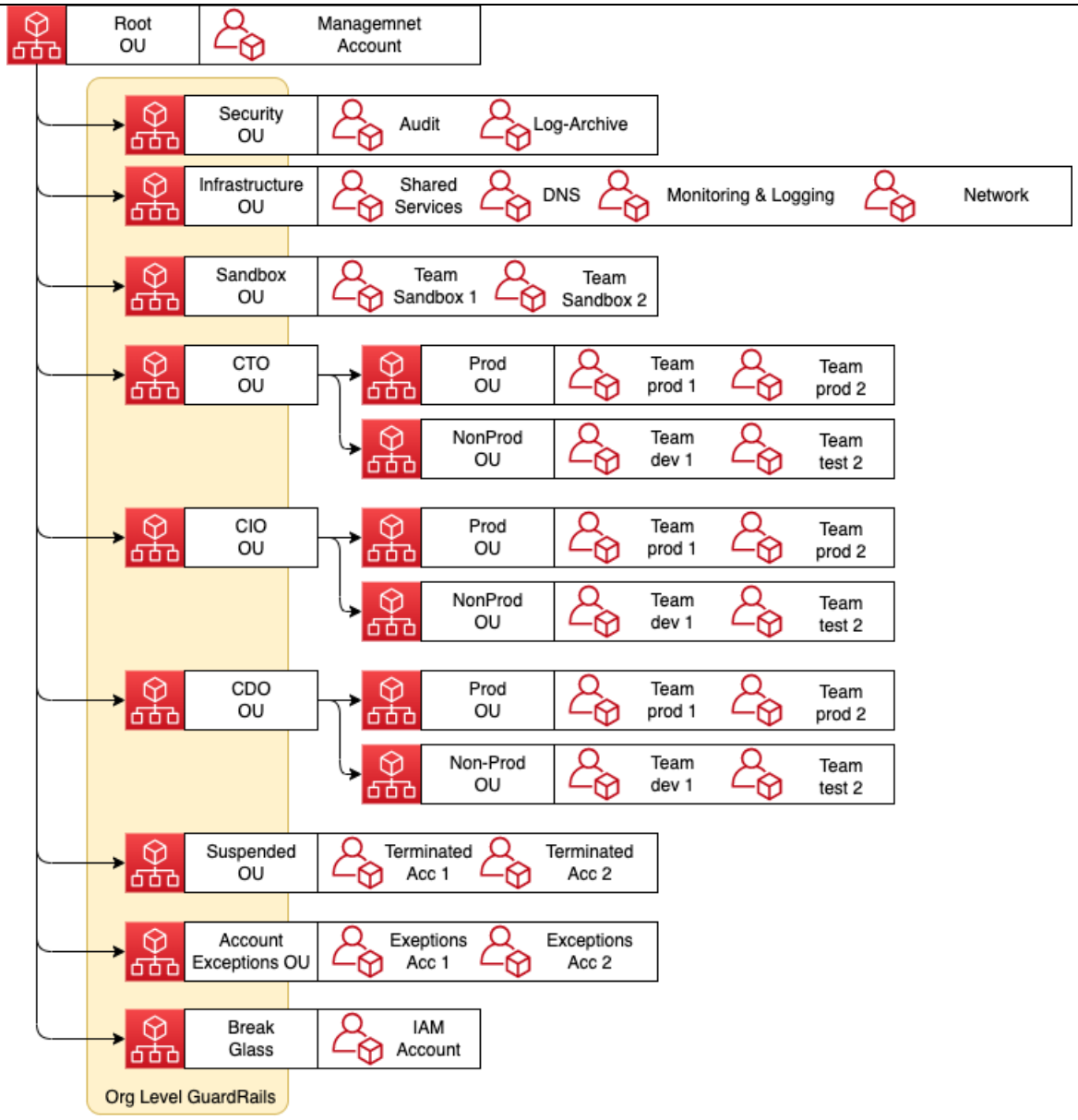


FIGURE 1- VMO2 ACCOUNT STRUCTURE

2.1 Root OU - Management Account

The Management Account is the account that gets created specifically for the new landing zone. This account will be used for consolidated billing for everything in the new landing zone. It may also be used for provisioning new AWS accounts, managing AWS Organizations Units & guardrails and managing user access & permission set through AWS Single Sign-On. Cross account roles will be used to provide access to this account from other platform accounts where provision or billing access is needed. This OU will not run any workload considered as non-management related.

2.2 Security OU

This Organizational Unit is used to apply controls to accounts that have global landing zone privilege and focus on security and compliance for the entire landing zone. Separation of concerns (SoC) is a design principle for separating privilege into discreet accounts to allow specific users or functions in those accounts to perform specific actions, the VMO2 platform and Digital Security engineering teams will have access to these accounts, while it's expected that platform operations team will get granted access to generated logs. The table below identifies some common/optional accounts that are typically found under this OU.

Account	Purpose	Rational
Audit/Security	The audit account is a restricted account that's designed to gives the security and compliance teams read and write access to all accounts in the landing zone. From the audit account, programmatic access to review accounts, by means of a role that is granted to Lambda functions is possible. The audit account does not allow a user to log in to other accounts manually.	Separate the audit, analysis and control of the landing zone from the use of the landing zone. Group and BU audit functions can be hosted here
Logs Archive	The Logs Archive account works as a repository for logs of API activities and resource configurations from all accounts in the landing zone.	Separate the storage, analysis and management of logs from the source of the logs
IAM (3rd party)	The IAM account is where any none SSO IAM accounts are created for general usage such as limited 3rd party access. These accounts must be added manually to this OU	Separate 3rd party access and cross account roles users who are not members of the corporate identity platform (Microsoft AD, Google, Okta etc.)
Security Services	This account hosts any 3rd party security software for logging, antivirus etc. This account can be embedded on the Audit account if there is separation of duties.	Separate accounts to segment security applications

The last two accounts are mentioned as usual best-practices to consider.

Decision	VMO2 will not be using these initially. This decision can be reverted at any time.

2.3 Break Glass Account

Regular access to AWS accounts within the organization should be provided using federated access . The use and creation of IAM users is highly discouraged with the exception of break glass users. It is generally recommended that organizations create at least 2 IAM users to prevent lock down in case one of them is not available, and up to 4 IAM users based on the size of your organization.

We recommend these users be configured with a hardware-based MFA device, which would be used in exceptional circumstances to gain access to the organization management account or sub-accounts within the organization by assuming a role. The use cases for break glass access include:

- Failure of the organization'sIdP.
- A security incident involving the organizations'IdP(s).
- A failure IAM IdentityCenter.
- A disaster involving the loss of an organization's entire cloud or IdP teams. It is important that access to these roles is monitored, and alarms and alerts are triggered when the roles are used to access the environment.

It is worth highlighting that AWS Organizations Service Control Policies do not apply to the organization management account, and access to this account would grant admin status to the entire organization, given the trust relationship to the management account. Therefore, access to break glass IAM users must be tightly controlled, yet accessible via a predefined and strict process. This process often involves one trusted individual having access to the password and a different trusted individual having access to the hardware MFA key, meaning it typically requires two people to access any one set of break glass credentials.

Why is this account needed?

The organization management account is used to provide break glass access to AWS accounts within the organization. Break glass (which draws its name from breaking the glass to pull a fire alarm) refers to a quick means for a person who does not have access privileges to certain AWS accounts to gain access in exceptional circumstances, using an approved process.

Refer to this link for additional information (Break Glass Access)

Decision	VMO2 will use Break Glass account.

2.4 Shared Services OU

This Organizational Unit is used to apply controls to accounts that will contain AWS accounts for hosting various platform operator "Services". Services on this OU, like Cloud DNS, are commonly used by the accounts deployed at the organization level. They are grouped when they are common and shared. For example, the TGW in Network Account is shared with several other accounts. The table below identifies some common accounts that are typically found under this OU.

The services defined into the Shared Services OU are designed to be shared on purpose between OUs and accounts, avoiding duplicated implementations and unnecessary costs. It's also possible to individualize them if it's a hard design requirement from VMO2, but this decision will bring associated additional costs that can be avoided.

Account	Purpose	Rational
Networks	This account is where direct connect and transit gateway are managed from	Separate TGW and DX management from other accounts. Please see the Network design section for more detail.
Common Services	This account will host common services such as AWS SSM, Base AMIs to be shared with rest of the accounts, Image Builder service, Central ECR repository, etc.	Support common cloud services, provided by DevOps teams.
DNS	This account hosts internal DNS zones	Support global LZ DNS resolution.
Monitoring & Logging	These accounts host 3rd party or open-source logging, monitoring and tracing tools	Standardized monitoring and tracing tools
Networking Egress(*)	The Networking Egress account will host services like Transparent Proxy, NAT etc.	Separate the control of egress traffic from the resources.
Networking Ingress(*)	This account will be hosting all the ingress networking components of the Landing Zone, including gateway for terminating office/developer VPN's and ingress proxies.	Separate the control of ingress traffic from the resources.

(*) The last two accounts are included to show the complete options deployed in a cloud environment where these capabilities are not present. If this capability is implemented outside the cloud platform (on-prem) or by other means those accounts can be obviated.

Decision	VMO2 will use Shared Services OU for DNS and Networking. Under consideration CommonServices, Egress/Ingress and Monitoring and Logging. Not planned at this stage for the first iteration of MVP.

2.5 Playgrounds/Sandboxes

This Organizational Unit is used to apply controls to accounts that's are used by teams or individuals to test and develop new ideas based on AWS services. Limited technical controls are applied to these accounts but cost control is very important and therefore accounts will have string cost controls and/or reporting.

2.6 CxO OUs

2.6.1 Non-Production

This Organizational Unit is used to apply controls to accounts that's are used by teams to build and test their applications. The accounts can host individual team build/test tools and resources as well as integration landscapes

that's are used by multiple teams to test production like configurations. Production controls will be applied but may not be enforced and exceptions alerted on. Dev and Test accounts are usually considered under this OU.

2.6.2 Production

This Organizational Unit is used to apply controls to accounts that's are used by teams to host production instances. The accounts will host production landscapes that's are used by multiple teams to host their production services and/or individual team production accounts. Production controls will be applied and enforced as Accounts in this OU represent the final step on the development lifecycle where applications are running in production.

2.6.3 Staging

This Organizational Unit is used to apply controls to accounts that are used by a single team or teams to perform integration testing. Accounts in this OU represents the 2nd step on the development lifecycle where deployments can be tested in production-like environments with other systems running in AWS, on-premise or in a SaaS deployment. The rational for this OU is that it allows polices to be applied to staging environments that align with production controls.

Decision	VMO2 will not use the Staging OU. The associated loads will be absorbed and managed on development by the Non-Production OU associated accounts.

2.7 Exceptions OU (optional)

This Organizational Unit is created if it's needed. Used to apply controls to accounts that are used by a single team or teams of developers to host accounts that don't/can't comply with any of the other OU's. The rational for this OU is that it allows a limited set of policies to be applied to accounts that's have been migrated into the landing zone and need some time for risks to be mitigated or for accounts for 3rd party solutions that cannot comply with production controls etc.

2.8 Suspended OU

This Organizational Unit is used to apply controls to accounts that are being deleted/suspended. The rational for this OU is that it allows accounts be isolated from other accounts while they are in the process of being deleted and/or change policies that may prevent deletion such as allowing the root account to perform actions such as requesting suspension to changing passwords. When an account is required to be closed, the account is kept in suspended state by 90 days, so it can be reactivated after closing if it's required.

2.9 Incident Response OU (optional)

This Organizational Unit is used to apply controls to accounts that are being used to investigate a security incident by forensically studying the payload using a sandbox/quarantine account and forensic tools located in the same account or a separate forensic account. The rational for this OU is that it allows accounts be fully isolated from both a policy and network perspective to allow dangerous malware to be studied and has separate control imposed to fully isolate it from the rest of the organization.

This account is different from the Suspended OU, where the OUs are parked until its final closing after a period of time (90 days by default). You cannot operate on the suspended account, while you have full access to the Incident Response OU which is a regular OU at the end.

Decision	VMO2 will not implement this OU initially. Its creation can be done at a later stage.

2.10 Account Naming Pattern for VMO2

The name of resources becomes ever more important as you begin to automate, having an account name that can be used to determine key operational controls and also makes it easy for users to understand where they are in the AWS cloud is critical. The pattern in use is shown below and it cannot exceed 50 characters.

{{AppName}}-{{Area}}-{{Environment}}-{{Number}}

Account name **should not** contain the **region names**.

AppName	A friendly identifier for the application landscape or vmo2	digital, analytics, etc. / vmo2
Area	the OU the account will be deployed into and therefore the control point for policy, this can be represented as an acronym.	cloud / cto / cio / cdo
Environment	A predefined identifier of environment	p = prod n = non-prod s = shared
Number	A simple 3 digit assigned to that account to make it unique	001-999

Some examples of this naming standard are shown below but the value from this approach is that it allows the users to identify their account in an easy-to-use manner **{{TeamName}}-{{TeamEnvironment}}** and for the automation to identify the correct OU and network controls to deploy into using the **{{OUType}}** elements which can be propagated by the automation or manually entered by the provisioning team. ID is just a 3-digit code for uniqueness.

1. **Example 1:** *digital-cdo-p-012*
2. **Example 2:** *cloudsvcs-cto-n-007*
3. **Example 3:** *vmo2-cdo-p-001*

You can create resources adding a friendly name up front of the account name.

2.10.1VPC Naming Pattern for VMO2

The VPC pattern is shown below and it cannot exceed 50 characters.

{{ServicePrefix}-{{Region}}-{{Environment}}-{{Number}}

Prefix	Service Prefix (VPC)	vpc
Region	AWS Region where the VPC is located	London (eu-west-2) → ew2 Frankfurt (eu-central-1)→ec1
Environment	A predefined identifier of environment	p = prod n = non-prod s = shared
Number	A simple 3 digit assigned to that account to make it unique	001-999

Examples:

1. **Example 1:** *vpc-ew2-p-001*
2. **Example 2:** *vpc-ew2-s-007*

2.10.2VPC Subnet Naming Pattern for VMO2

The VPC subnet pattern is based on the VPC naming but adding additional patterns. It cannot exceed 50 characters. Definition and examples to follow.

{{ServicePrefix}-{{Region/AZ}}-{{Environment}}-{{SecurityLevel}}-{{Number}}

Prefix	Service Prefix (subnet)	sn
Region/AZ	AWS Region/AZ where the VPC is located AZ= a,b,c, etc.	London (eu-west-2) → ew2 Frankfurt (eu-central-1)→ec1
Environment	A predefined identifier of environment	p = prod n = non-prod s = shared
Security Level	Subnet Security Level	pub = public prv = private mgt = management
Number	A simple 3 digit assigned to that account to make it unique	001-999

Examples:

1. **Example 1:** *sn-ew2a-p-pub-001*
2. **Example 2:** *vpc-ew2b-s-prv-007*

2.10.3EC2 Naming Pattern for VMO2

The EC2 pattern is based on following pattern. It cannot exceed 50 characters. Definition and examples to follow.

{{CloudPlatform}+{{Environment}}+{{App+Role}}+{{Region}}+{{Number}}

CloudPlatform	Cloud Platform Service Prefix	Aws
Environment	A predefined identifier of environment	p = prod n = non-prod s = shared
Application	Application name	peg = pega
Role	Role name	web = web server app = application server dba = database server
Region	AWS Region/AZ where the VPC is located AZ= a,b,c, etc.	London (eu-west-2) → ew2 Frankfurt (eu-central-1)→ec1
Number	A simple 3 digit assigned to that account to make it unique	001-999

Examples:

1. **Example 1:** *aws+p+peg+web+ew2+001 → awsppegwebew2001*

2.10.4Security Groups (SGs) Naming Pattern for VMO2

The SGs pattern is based on following pattern. It cannot exceed 50 characters. Definition and examples to follow.

{{Prefix}+{{Region}}+{{Environment}}+{{App+Role}}+{{Number}}

Prefix	Prefix Name	sg
Region	AWS Region/AZ where the VPC is located AZ= a,b,c, etc.	London (eu-west-2) → ew2 Frankfurt (eu-central-1)→ec1
Environment	A predefined identifier of environment	p = prod n = non-prod s = shared
Application	Application name	peg = pega
Role	Role name	web = web server app = application server dba = database server
Number	A simple 3 digit assigned to that account to make it unique	001-999

Examples:

1. **Example 1:** *sg+ew2+p+peg+web+ 001 → sgew2ppegweb001*

2.11 Email address for AWS accounts

At the onboarding procedure we have to minimize amount of unique email and carefully plan them for new AWS accounts, like logically group into same email. Currently, within AWS environment it is not possible to create multiple accounts with the same email. For the accounts we use the same email addresses with different postfixes after a + symbol.

The following pattern to be used for email addresses:

{{Prefix}}{{OUtype}}{{TeamName}}{{+Postfix}}@virginmediao2.co.uk

This email needs to exist before account creation. Both aliases will forward the mails to this account.

- Prefix – VMO2Cloud
- OUType (Organization) – CTO, CDO, CIO
- TeamName – freeform (e.g. TeamA)
- Postfix – name + id (000) – (e.g. prod-001)
- **@virginmediao2.co.uk**

VMO2CloudCTO**TeamA**+prod-001**@virginmedia02.co.uk** - email for TeamA on CTO/production OU.

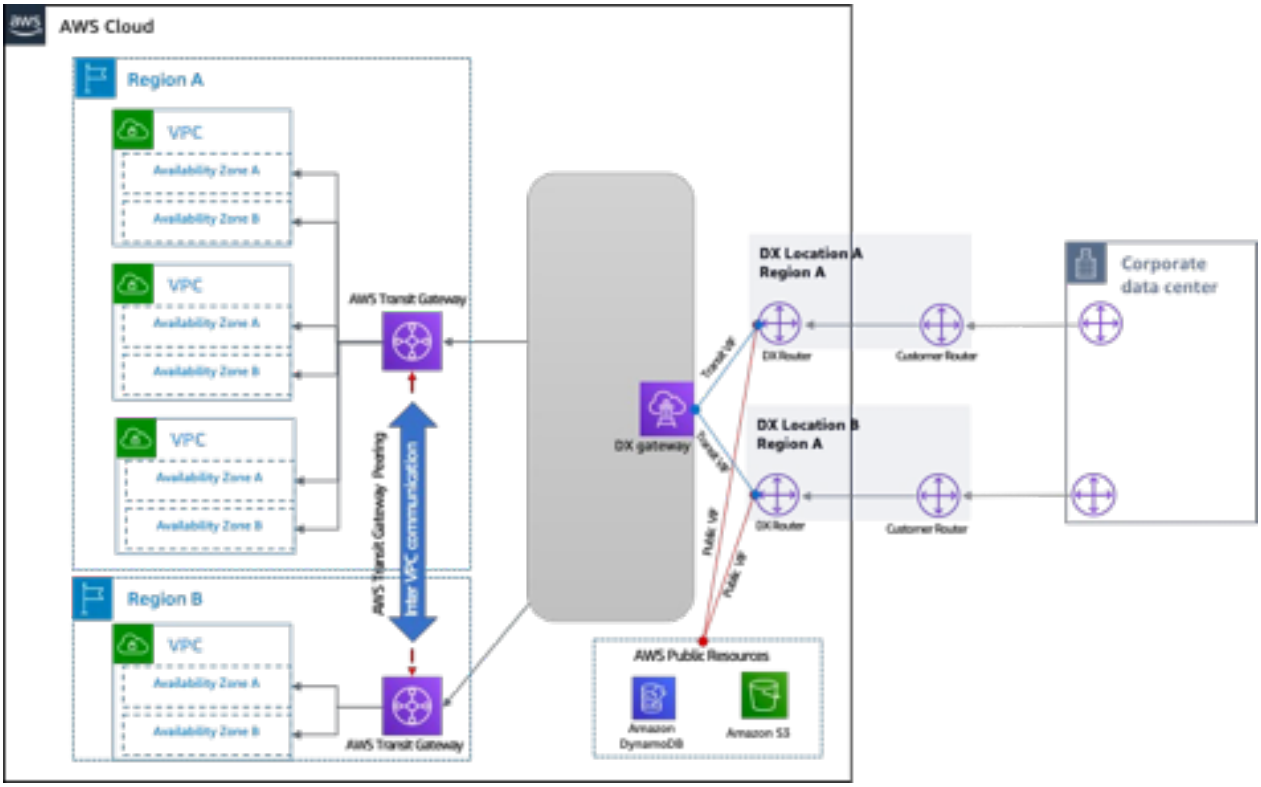
2.12 Landing Zone Accounts and Emails

The LZ accounts created with Control Tower that have been listed above as constituents of the LZ platform will have the following format vmo2-{{AccountName}}, and the email will be aligned with the name without hyphens in the middle, and with first letter capital

Account	AWS Name	Email	Purpose
Management	vmo2-management	VMO2CloudManagement@virginmediao2.co.uk	Root and Payer Account
Security OU	vmo2-logarchive	VMO2CloudLogAudit@virginmediao2.co.uk	Organization Central Audit Logging Account
	vmo2-audit	VMO2CloudAudit@virginmediao2.co.uk	Audit Account
Infrastructure	Network	Tba (to be added later)	
	Shared Services	Tba (to be added later)	
	DNS	Tba (to be added later)	
	Monitoring & Logging	Tba (to be added later)	

2.13 Landing Zone Product Development

The Landing Zone itself should be considered a software release itself with a heavy focus on infrastructure as code (IAC), versioning and regular update/deployment cycles. In order to facilitate this approach, VMO2 is following a Trunk-based development. On this development process, there is a version control management practice where developers merge small, frequent updates to a core "trunk" or main branch streamlining the merging and integration phases allowing CI/CD while it increases software delivery and organizational performance. Is usually combined with feature flags, to decouple deployment from release, so the features are deployed when they are ready without any delay.



While this is a valid approach, articulate a mechanism to safely test certain changes/additions like guardrails (service control policies, config rules etc.) and new or updated shared or core services, etc. impacting the organization:

- Testing and automating new OU structure
- Test and automating account creation, configuration and deletion process/code
- Testing and automating guardrails (service control policies, config rules etc.)
- Eventually, testing and automating the deployment and upgrade of any 3rd party/operations tools used to managed non-production and production in main organization.

Decision	VMO2 will not implement the approach based on two parallel organizations: test and prod OU. It will keep using the trunk-based approach on the prod organization only.

The non-Prod OU can also be used for specific testing of Service Control Policies (SCP's) and AWS Config rules. The group owning the Landing Zone design provides a channel for users to communicate back enhancements and bug fixes including addition SCP's or Config rules that have been developed in various markets.

Decision	VMO2 will have both Prod and Test under the Non-Prod OU. It will not use a Test OU. MVP will use Prod Organization and the IaC pipeline already being developed to cater for branching for Test Org.

2.14 MVP Organization Estimated Costs

The table below gives a high-level estimated cost of what a new Organization would cost. The main use is for testing polices which are generally free, apart from AWS Config. It is also assumed some of the security prototyping for Security Hub and Inspector would also happen in this org as part of the security design. Any resource services should be used in an *adhoc* manner and deleted after use and would mostly use new account free tier incurring no costs.

Account	Resource	Cost (monthly/USD)
Root	New master payer account	$0
Root	Organization & OU's	$0
Root	SCP	$0
Root	Tagging Polices	$0
Multiple	Config	Based on - London region - 200 Config items - 1000 rules/eval - 1000 conformance packs **$8.20**
Non-Production child account	Account	$0
Non-Production child account	Test EC2 Workload	Free Tier $0
Non-Production child account	Inspector	Based on: - 2 EC2 instances - 100 account checks - 100 finding ingested per account **$0.50**
Production child account	Account	$0
Production child account	Test EC2 Workload	Free Tier $0
Security Account	Security Hub	Based on - 5 child accounts - 2 ECR images/20 re-scans - 2 ECR scan on push **$3.86**
estimated monthly costs		**$12.56**

cost provided by AWS Pricing calculator based on London region and on-demand pricing

3 Networking

The proposed AWS Network Architecture is based on the following design tenants:

- Alignment of network architecture with Account architecture.
- Alignment with Security Reference Architecture recommendations (security section).
- Minimize the number of VPCs and reduce operational effort.
- Consider support of IP dual/stack support for future deployments.
- Align TGW route tables with OU structure to ensure a consistent way of managing AWS guardrails and network controls.
- Transit gateway (TGW) will be the central point of IP connectivity. It will support:
    - Inter-OU routing
    - AWS Cloud to On-Prem and On-Prem to AWS Cloud
    - DNS traffic towards/from On-Prem
    - Egress traffic, if it's considered to do it via TGW for selected account(*)
- Architecture will support implementation to address VMO2 requirements disclosed at this time.

(*) Implementation of dedicated Egress Point for selected accounts (Prod accounts, for example) via a central Egress Accounts is under discussion and not decided at this point in time.

Decision	VMO2 has decided not to consider the Egress Account, but this decision can be reverted later if it's needed for service/network evolution or customer requirements.

3.1 Networking Decisions

In order meet the previous goals, the following **decisions** have been made to guide the technical implementation. Note the term "Managed" is used when there is an IaC process to deploy those resources.

Decision	This process will be owned by the Platform Engineering Team, not the Account/OU owners.

Other decisions from Networking perspective:

1. We will consider the following traffic classes:

    - **MANAGED -** IP Ranges that are routable on AWS from the On-Prem network and ranges from On-Prem routable from AWS Networks.
        - **Routable** – ranges reachable from On-Premise and able to reach On-Prem network.
            - **DX** - ranges exclusively used to reach the DX. Restricted to Direct connection routing.
            - **Private -** Ranges on the **10.46.0.0/16** range.
        - **Non-Routable** -
            - **Private -** Ranges on the **100.64.0.0/16** of use only on the cloud environment. Allow routing between the accounts and it can be used to support the TGW Attachments and additional private subnets. Non-routable to the outside world.
    - **Non-MANAGED** – IP ranges under control of customers. Customers are responsible from its assignment and use. The Scope of those ranges is the Account, so we can have overlapping IP addresses ranges between different accounts.
        - **Routable**
            - **Public** - Reachable from internet via IGW/NatGW. Not reachable from On-Premise.
            - Require to use EIP on the services exposed. Not a Range but punctual public access.
        - **Non-Routable**.
            - **Private** - to be used internally to the account. No access to other accounts, because not routed via TGW. Non-routable to/from On-Prem/Internet
            - Any range with local scope, ideally on the RFC1918.

Decision	DX range can be used by Production and Non-Production account. Sandbox will not have access to this range. Private Range for TGW Attachments can use a minimum of /28 netmask.

Option to be considered later at any point – **Managed Shared Public** - IP ranges controlled centrally by the platform team and shared with the OUs. The sharing can happen at account, OU or Organization level.

This is an optional range can be deployed centrally from the Network account as a shared resource via RAM (Resource Access Management). It's recommended to keep a subrange on the 10.46.0.0/16 for this purpose. Allow interOU / InterAccount Communication without passing traffic via TGW. It can hold IGW/NatGW for all the participant accounts.

Decision	The usage of Public Routed Range is not initially part of VMO2 MVP.

2. Isolation between operational and application functions will be achieved through platform controls such as IAM, security groups and NACLs.
3. Traffic Inspection between different classes will **not** be considered initially.
4. Network Firewall is **not** considered initially.
5. IP address separation between OUs, production and non-production must exist. Controlled via VPC ranges and Routing tables on the TGW.
6. Dual-stack IPv4/v6 will not be deployed initially. It's not a requirement and it can be added later on demand
7. . Hybrid DNS implementation is required.
8. AWS services will be exposed through private endpoints but centralize those endpoints into a common account under Shared Services OU is on hold at the moment.
9. TGW will be deployed in a network account and shared with the organization.

Decision	Traffic inspection is not a requirement on the MVP. It will rely on SGs (ENI) and NACLs (subnets). Not Firewall (centralized or distributed) will be implemented. No internal traffic analysis. IPv6 not deployed initially. It can be a requirement later. Not gating factors. The usage of VPCe's is under discussion. Not included on the current phase. Shared TGW will be used. No Shared VPCs. Egress services will not be exposed centralized. Ingress not considered at this moment.

This document uses some well understood network patterns described as its foundation and listed for reference purposes here:

- Centralized access to VPC Endpoints
- Centralized Egress
- AWS PrivateLink
- AWS Traffic Mirroring

3.2 Proposed Network Reference Architecture

The following figures illustrate the proposed network reference architecture, which is described in detail in the remainder of this section, the Transit Gateway is owned by the Shared Services OU/ Network Account and shared by RAM within the organization. This will allow to every account on the organization create attachments and request acceptance, so the automation implemented by the Platform team can authorize it and associate it to the right Routing Table.

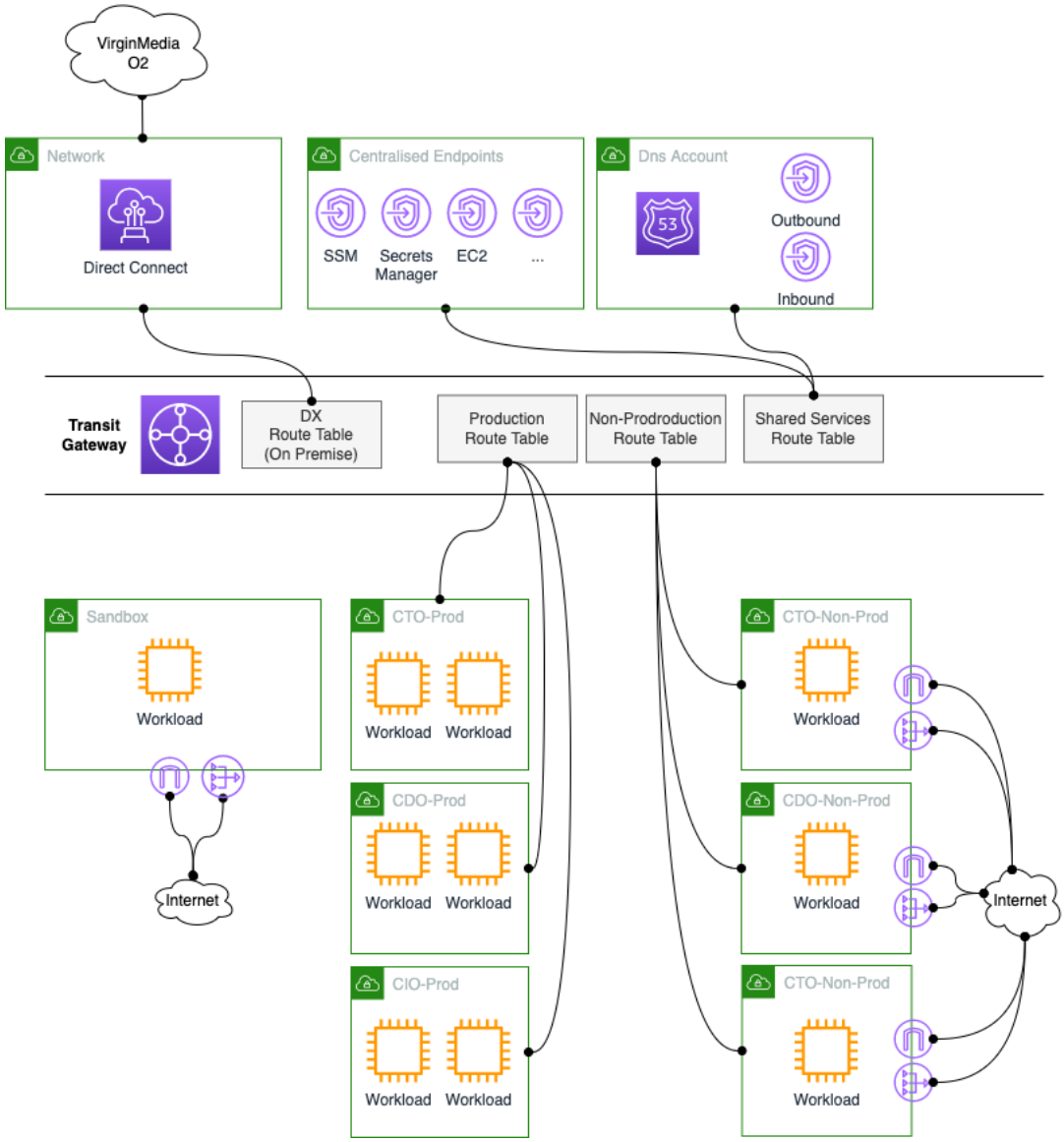


FIGURE 2-GENERAL VMO2 NETWORKING ARCHITECTURE

3.3 IP addressing

3.3.1 IPv4 related considerations

The size of the network to be implemented will use IPv4 ranges that will be divided based on three IP addressing categories: Small, Medium and Large. Each of those ranges can be used without further considerations beyond the type of traffic that the IP range is supporting for routing configuration and the size of the masking being used.

Since there is no indication about expected numbers of subnets, hosts, accounts, etc., for the MVP we will made some assumptions to illustrate how the networking IP addressing can be implemented.

DX Routing has assigned ad 10.46.0.0/16 superblock, that need to be further divided to accommodate the different VPC sizes (S,M,L) mentioned. The mechanisms and decisions under consideration for the split of those is out of scope of this document at this point in time, but we can do an exercise on a possible division under certain premises:

- Consider 3 types of ranges (S, M & L).
- Assume at least coverage of 3 AZs with Symmetric IP Ranges for S and M.
- Large Ranges can be managed on demand due to its nature to avoid unnecessary reservations.

The figure below shows an example of a possible split.

Subnet address	Netmask	Range of addresses	Useable IPs	Hosts	Divide
10.46.0.0/22	255.255.252.0	10.46.0.0 - 10.46.3.255	10.46.0.1 - 10.46.3.254	1022	Divide
10.46.4.0/22	255.255.252.0	10.46.4.0 - 10.46.7.255	10.46.4.1 - 10.46.7.254	1022	Divide
10.46.8.0/22	255.255.252.0	10.46.8.0 - 10.46.11.255	10.46.8.1 - 10.46.11.254	1022	Divide
10.46.12.0/22	255.255.252.0	10.46.12.0 - 10.46.15.255	10.46.12.1 - 10.46.15.254	1022	Divide
10.46.16.0/22	255.255.252.0	10.46.16.0 - 10.46.19.255	10.46.16.1 - 10.46.19.254	1022	Divide
10.46.20.0/22	255.255.252.0	10.46.20.0 - 10.46.23.255	10.46.20.1 - 10.46.23.254	1022	Divide
10.46.24.0/22	255.255.252.0	10.46.24.0 - 10.46.27.255	10.46.24.1 - 10.46.27.254	1022	Divide
10.46.28.0/22	255.255.252.0	10.46.28.0 - 10.46.31.255	10.46.28.1 - 10.46.31.254	1022	Divide
10.46.32.0/24	255.255.255.0	10.46.32.0 - 10.46.32.255	10.46.32.1 - 10.46.32.254	254	Divide
10.46.33.0/24	255.255.255.0	10.46.33.0 - 10.46.33.255	10.46.33.1 - 10.46.33.254	254	Divide
10.46.34.0/24	255.255.255.0	10.46.34.0 - 10.46.34.255	10.46.34.1 - 10.46.34.254	254	Divide
10.46.35.0/24	255.255.255.0	10.46.35.0 - 10.46.35.255	10.46.35.1 - 10.46.35.254	254	Divide
10.46.36.0/24	255.255.255.0	10.46.36.0 - 10.46.36.255	10.46.36.1 - 10.46.36.254	254	Divide
10.46.37.0/24	255.255.255.0	10.46.37.0 - 10.46.37.255	10.46.37.1 - 10.46.37.254	254	Divide
10.46.38.0/24	255.255.255.0	10.46.38.0 - 10.46.38.255	10.46.38.1 - 10.46.38.254	254	Divide
10.46.39.0/24	255.255.255.0	10.46.39.0 - 10.46.39.255	10.46.39.1 - 10.46.39.254	254	Divide
10.46.40.0/24	255.255.255.0	10.46.40.0 - 10.46.40.255	10.46.40.1 - 10.46.40.254	254	Divide
10.46.41.0/24	255.255.255.0	10.46.41.0 - 10.46.41.255	10.46.41.1 - 10.46.41.254	254	Divide
10.46.42.0/24	255.255.255.0	10.46.42.0 - 10.46.42.255	10.46.42.1 - 10.46.42.254	254	Divide
10.46.43.0/24	255.255.255.0	10.46.43.0 - 10.46.43.255	10.46.43.1 - 10.46.43.254	254	Divide
10.46.44.0/24	255.255.255.0	10.46.44.0 - 10.46.44.255	10.46.44.1 - 10.46.44.254	254	Divide
10.46.45.0/24	255.255.255.0	10.46.45.0 - 10.46.45.255	10.46.45.1 - 10.46.45.254	254	Divide
10.46.46.0/24	255.255.255.0	10.46.46.0 - 10.46.46.255	10.46.46.1 - 10.46.46.254	254	Divide
10.46.47.0/24	255.255.255.0	10.46.47.0 - 10.46.47.255	10.46.47.1 - 10.46.47.254	254	Divide
10.46.48.0/20	255.255.240.0	10.46.48.0 - 10.46.63.255	10.46.48.1 - 10.46.63.254	4094	Divide
10.46.64.0/19	255.255.224.0	10.46.64.0 - 10.46.95.255	10.46.64.1 - 10.46.95.254	8190	Divide
10.46.96.0/19	255.255.224.0	10.46.96.0 - 10.46.127.255	10.46.96.1 - 10.46.127.254	8190	Divide
10.46.128.0/17	255.255.128.0	10.46.128.0 - 10.46.255.255	10.46.128.1 - 10.46.255.254	32766	Divide

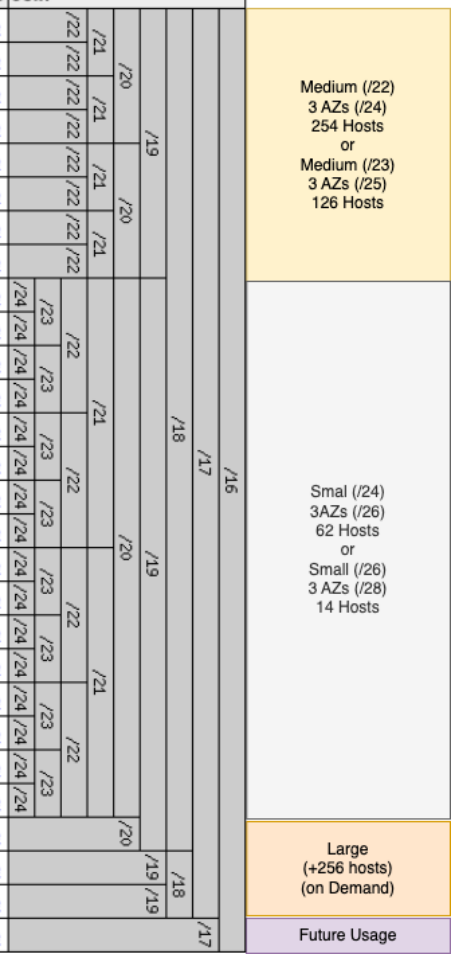


FIGURE 3 – IP RANGE SPLIT EXAMPLE

To illustrate the previous example, let's use the Medium Range case. The assumption is considering a need to deploy any service in at least 3AZs on the London Region. For doing that, we need to split the medium range (/22) in three, so at least we will use 2 more netmask bits which lead us to a /24 for each AZ with 254 hosts. If we consider /23 for the Medium Range, then /25 give us 3 AZs with 126 hosts each. For the small range we follow similar approach /24 to give 3AZs of /26 (62 hosts) or /26 to give us 3AZs of /28 (14 hosts).

In the previous example we are leaving a full /17 range for future usage to be divided with similar criteria. The Large ranges are accommodated at the end. Since those are usually an exception, we don't want to do any assumption about number of AZs and Host x AZ to avoid unnecessary IP reservation. As an example, we could consider the two /19, to be divided in 3AZs which lead us to /21 Ranges for Large Hosts with 2046 hosts on each group.

10.46.48.0/20	255.255.240.0	10.46.48.0 - 10.46.63.255	10.46.48.1 - 10.46.63.254	4094	Divide
10.46.64.0/21	255.255.248.0	10.46.64.0 - 10.46.71.255	10.46.64.1 - 10.46.71.254	2046	Divide
10.46.72.0/21	255.255.248.0	10.46.72.0 - 10.46.79.255	10.46.72.1 - 10.46.79.254	2046	Divide
10.46.80.0/21	255.255.248.0	10.46.80.0 - 10.46.87.255	10.46.80.1 - 10.46.87.254	2046	Divide
10.46.88.0/21	255.255.248.0	10.46.88.0 - 10.46.95.255	10.46.88.1 - 10.46.95.254	2046	Divide
10.46.96.0/21	255.255.248.0	10.46.96.0 - 10.46.103.255	10.46.96.1 - 10.46.103.254	2046	Divide
10.46.104.0/21	255.255.248.0	10.46.104.0 - 10.46.111.255	10.46.104.1 - 10.46.111.254	2046	Divide
10.46.112.0/21	255.255.248.0	10.46.112.0 - 10.46.119.255	10.46.112.1 - 10.46.119.254	2046	Divide
10.46.120.0/21	255.255.248.0	10.46.120.0 - 10.46.127.255	10.46.120.1 - 10.46.127.254	2046	Divide
10.46.128.0/17	255.255.128.0	10.46.128.0 - 10.46.255.255	10.46.128.1 - 10.46.255.254	32766	Divide

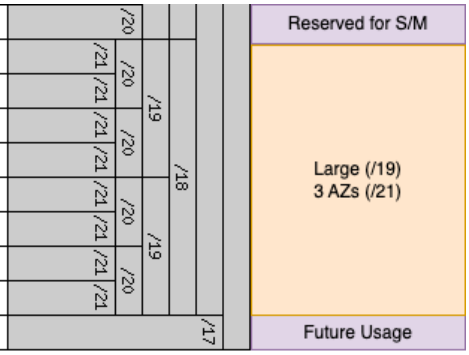


FIGURE 4 - LARGE IP RANGE EXAMPLE

Note – for each IP range assigned to a VPC, AWS always reserve for its own usage, the first 4 IP's and the last one on the Range assigned to the VPC. Let's see an example considering 10.0.0.0/24. In this case those are the IP's taken by the AWS VPC service:

- 10.0.0.0 – network address
- 10.0.0.1 – VPC router (gateway)
- 10.0.0.1 – IP address of internal DNS. It's always there and it's network address plus (referenced as +2 address or .2 in many places)
- 10.0.0.3 – Reserved for future usage
- 10.0.0.255 – Network Broadcast. Not used/supported by AWS, so reserved for that reason.

Those need to deducted from the hosts x Range, so a /28 with 14 hosts it has a net number of 9 to be used by the account team. Similar considerations for the remaining ones, you need to consider the nominal number of hosts minus 5.

From here, we will apply the previous consideration to go further on the VPC examples. We will use Medium range with 3 AZs, which means that a 10.46.0.0/22, will have 4 subnets with /24 enough to cover the 3 AZs.

- 10.46.0.0/24 – future usage
- 10.46.1.0/24 – AZ1
- 10.46.2.0/24 – AZ2
- 10.46.3.0/24 – AZ3

If we do the equivalent with small network and we use /24 as baseline for the VPC, this will giThis approach of begin with a /22 can give us a number of options, we can assign for the previous case small subnets size if the number of hosts is not enough with the /24. If we need more than IPs upfront or later, we can assign additional CIDRs up to 5 x VPC. Beyond that number of VPCs x Account we must increase the limit (default is 5).

The IPs on the cloud are used from the ranges defined on the RFC1918 and non-routable additional ranges.

- RFC1918 ranges:
  - 10.0.0.0/8 – (10.0.0.0 – 10.255.255.255)
  - 192.168.0.0/16 – (192.168.0.0 – 192.168.255.255)
  - 172.16.0.0/12 – (172.16.0.0 – 172.31.255.255)
- RFC6598 IANA reserved Shared Address Space
  - 100.64.0.0/10

VMO2 Action	We need to identify the specific network ranges on that list in use on the cloud area. We specified that 10.46.0.0/16 is the one use from 10.0.0.0/8, the same will be required for the 100.64.x.x, since it can be used by some of the customers already.

Decision	The local address space based on the RFC6598, Carrier Grade NAT, it will be used for all platform management, service management and secure services if those exists in other regions in the future. This will mean they are not accessible from on premise systems without a proxy or NAT Gateway

The following table describes the usage for medium (M) size networks of this local address assignment:

Example Prefix	Description
100.64.0.0/10	Is allocated across the whole AWS Landing zone for private addressing giving us 32 /16 networks that we can use between our OUs.
100.64.0.0/22	We can apply the same principle than before and use /22 of that /16 to be shared between the accounts, give us /24 for each AZ. For the TGW Attachment we can use the 4th network with /26 for simplicity or /28 which is the minimal we can use for that purpose.
100.64.0.0/24 (AZ1)	
100.64.1.0/24 (AZ2)	
100.64.2.0/24 (AZ3)	
100.64.3.0/24 (TGW attachment)	
100.64.4.0/22	This will be another block for another OU, and the process will follow up to 64 blocks. If at any point this become exhausted, we can take the next /16 block and we will have another 64 blocks available.
100.64.4.0/24 (AZ1)	
100.64.5.0/24 (AZ2)	
100.64.6.0/24 (AZ3)	
100.64.7.0/24 (TGW attachment)	

A full example is on the next figure.

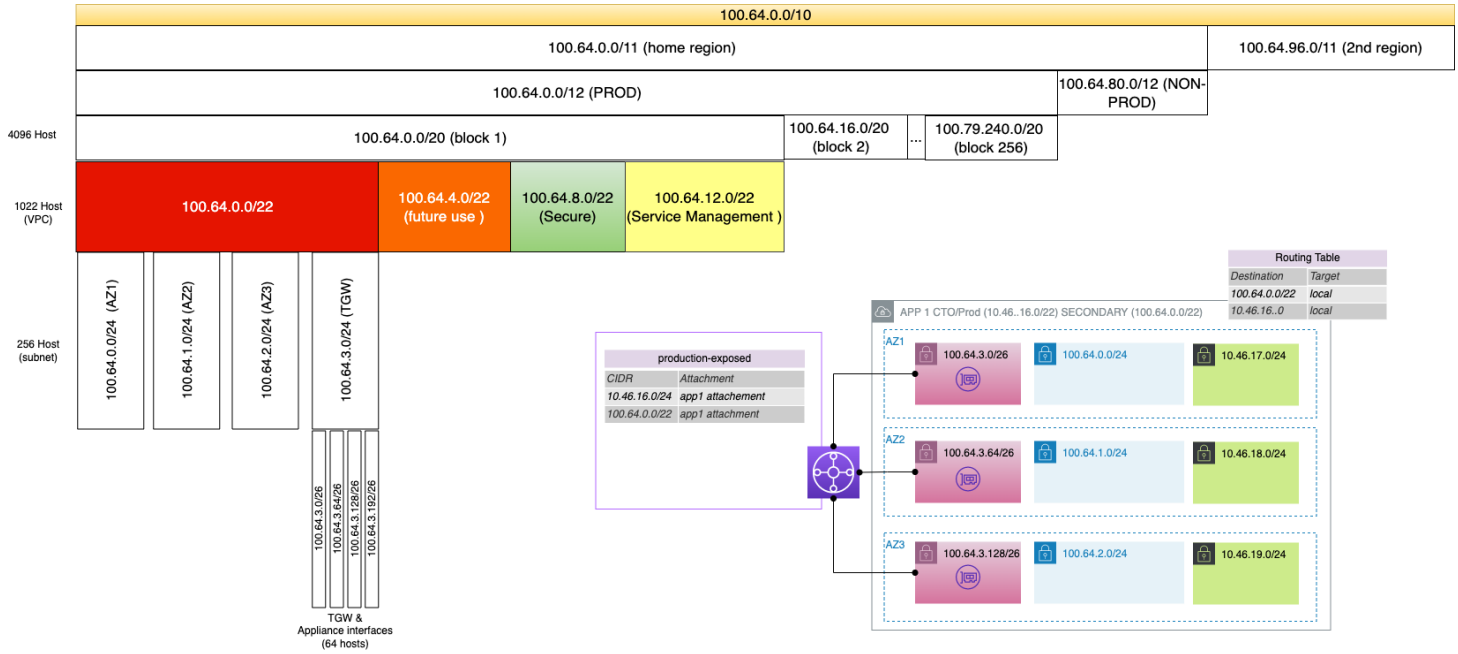


FIGURE 5 - RFC6598 SPLIT USE PROPOSAL

3.3.2 IPv6 related considerations

Although, there are no specific requirements related to IPv6 at network level, there are scenarios where the VPC hosting the applications may require the use of IPv6. In AWS, IPv6 address space assigned to VPCs is assigned from the *globally unique unicast* address range (2000::/3). There are two options for IPv6 address assignment:

- AWS-assigned IPv6 VPC classless inter-domain routings (CIDRs)
- Bring your own IPv6 CIDR Blocks (BYOIPv6)

In both cases, an IPv6 /56 assigned by AWS will be used, even if VMO2 will bring its own IPv6 range. Although there are no specific requirements for IPv6, the nodes that are part of the cluster will get IPv4 and IPv6 addresses from the ranges specified on the VPCs in case that some specific deployment require such setup.

Decision	No IPv6 will be used in the MVP initially. The capability is built-in on the AWS VPC implementation, so it can be added later on demand when needed.

3.3.3 Reserve Subnets CIDRs on Assignment

There is an option to exclude specific subnet/s from master subnet on each VPC is required to do so. AWS Documentation link below explains how to do that (https://docs.aws.amazon.com/vpc/latest/userguide/subnet-cidr-reservation.html)

3.4 Example of OU Addressing

The table below can be used as summary on the network split being considered.

(London) eu-west-2		CIDR	AZ1	AZ2	AZ3
CTO-Prod	Routable	10.46.0.0/22	10.46.1.0/24	10.46.2.0/24	10.46.3.0/24
	Local	100.64.0.0/22	100.64.0.0/24	100.64.1.0/24	100.64.2.0/24
TGW Att	Local	100.64.3.0/24	100.64.3.0/26	100.64.3.64/26	100.64.3.128/26
CTO Non-Prod	Routable	10.46.4.0/22	10.46.5.0/24	10.46.6.0/24	10.46.7.0/24
	Local	100.64.4.0/22	100.64.4.0/24	100.64.5.0/24	100.64.6.0/24
TGW Att	Local	100.64.7.0/24	100.64.7.0/26	100.64.7.64/26	100.64.7.128/26
Sandbox Public	Local	192.168.0.0/16	192.168.1.0/24	192.168.2.0/24	192.168.3.0/24
Sandbox Private	Local		192.168.11.0/24	192.168.12.0/24	192.168.13.0/24
Shared Services	Routable	10.46.236.0/22	10.46.236.0/24	10.46.237.0/24	10.46.238.0/24
	Local	100.64.236.0/22	100.63.236.0/24	100.63.237.0/24	100.63.238.0/24
TGWAtt	Local	100.63.239.0/24	100.63.239.0/26	100.63.239.64/26	100.63.239.128/26

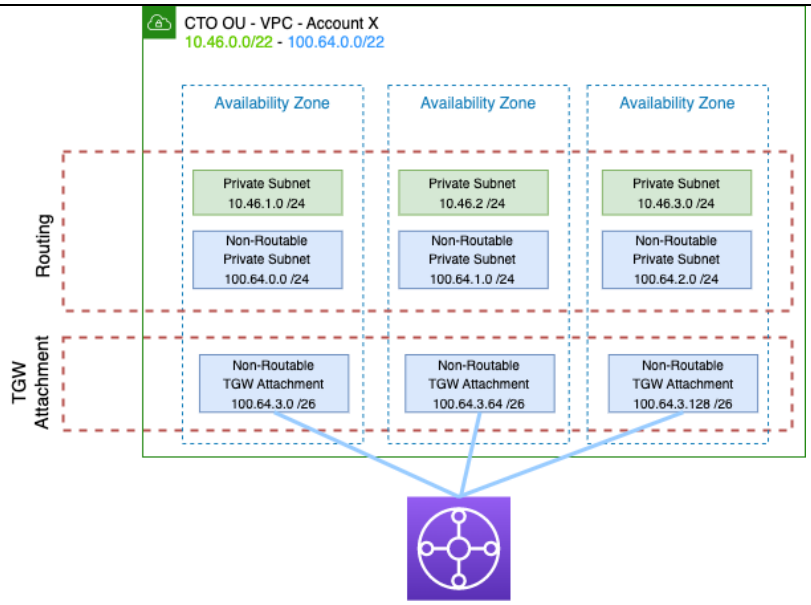


FIGURE 6- CTO PRODUCTION OU - ACCOUNT VPC EXAMPLE

Example – Sandbox (Note there is no TGW attachment). In principle Sandbox is an isolated environment from routing perspective for PoCs (proof of concepts or experimentation)

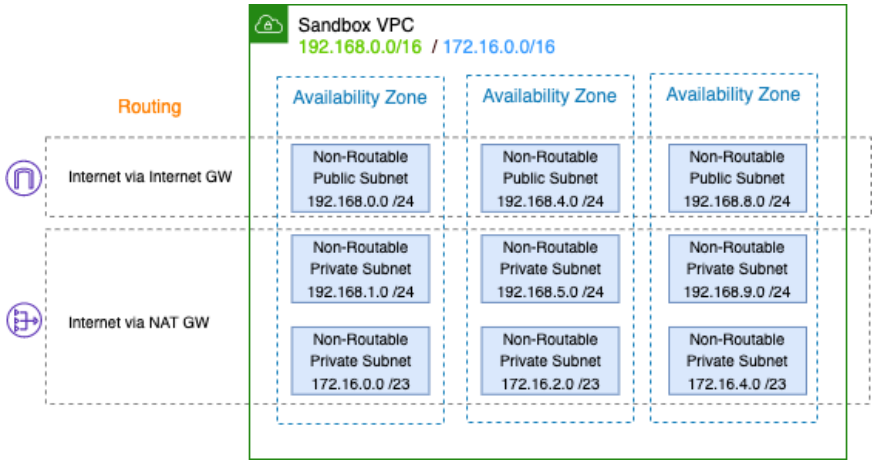


FIGURE 7- SANDBOX OU VPC EXAMPLE

Note – Control Tower has the capability to add a specific network to be used during the account creation proces. By default, this network is 172.31.0.0/16. This can be changed and/or deleted based on the VMO2 requirements.

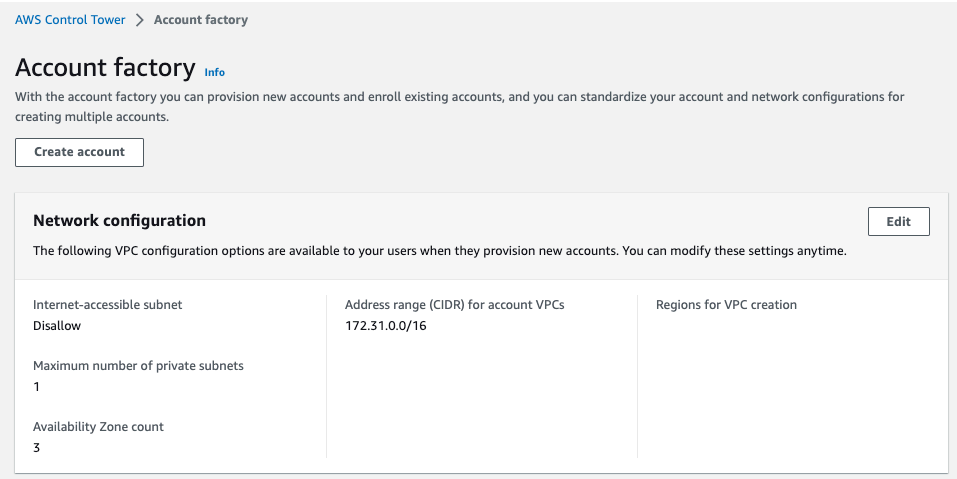


FIGURE 8 - CONTROL TOWER ACCOUNT FACTORY DEFAULT VPC

3.5 Connectivity between OU's/Accounts

AWS primarily offers two solutions for connecting accounts together: VPC peering and Transit Gateway.

A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account. The VPCs can be in different Regions (also known as an inter-Region VPC peering connection).

However, when we need to connect to on-premise, we need transitive connectivity and routing control TGW (transit gateway) is the preferred method. AWS Transit Gateway connects your Amazon Virtual Private Clouds (VPCs) and on-premises networks through a central hub. This connection simplifies your network and puts an end to complex peering relationships. Transit Gateway acts as a highly scalable cloud router—each new connection is made only once.

Decision	TGW will be used to managed On-Prem/Cloud connectivity and inter-VPC routing at Region level.

The use of the Transit Gateway to connect VPCs greatly simplifies the overall architecture, supports segmentation, allows to landing zone to scale to thousands of accounts and supports more advanced networking use cases. VPC peering will be used in very specific cases where there is a benefit of using it, due to the **low latency and higher bandwidth but there is no need to support transitive technology**, as such limited to specific application use cases.

3.5.1 Network Segmentation

Security best practices promote the use of controls at multiple layers, through the use of AWS Transit Gateway it is possible to achieve zoning / segmentation of VPCs. To ensure consistency in the security approach, it's intended to implement separation in accordance with the AWS Organization structure.

Through the use of different route tables on the Transit Gateway different routing behaviors can be achieved, this allows the Platform Team to control which environments have connectivity directly, through inspection layers or where communication should be blocked.

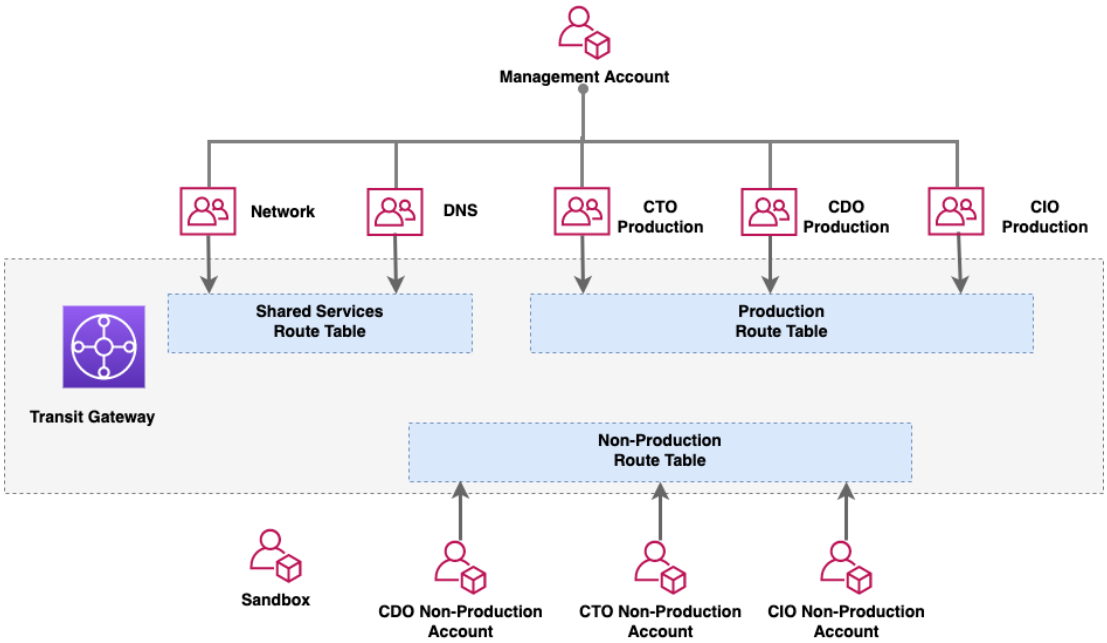


FIGURE 9 - OU GENERIC ROUTING SPLIT

Above, the TGW can be used to prevent the accounts in the production and development OU's from communicating with each other while allowing shared services to communicate with all networks.

3.6 Network Routing

To illustrate how the network routing works, we will consider two accounts on the CTO / Production OU and Non-Prod(eu-west-2). Each account has a single VPC with two subnets per availability zone, one for the workloads and one for the Transit Gateway (TGW) network interface (ENI). The routing table in the VPC is very simple, having the *local* VPC routes and a default route to the TGW. This means any traffic local to the VPC will stay in the VPC, all other traffic will be sent to the TGW.

For clarity we have considered some local VPCs that have scope only inside the Account VPCs and never get exposed to the remaining accounts via TGW.

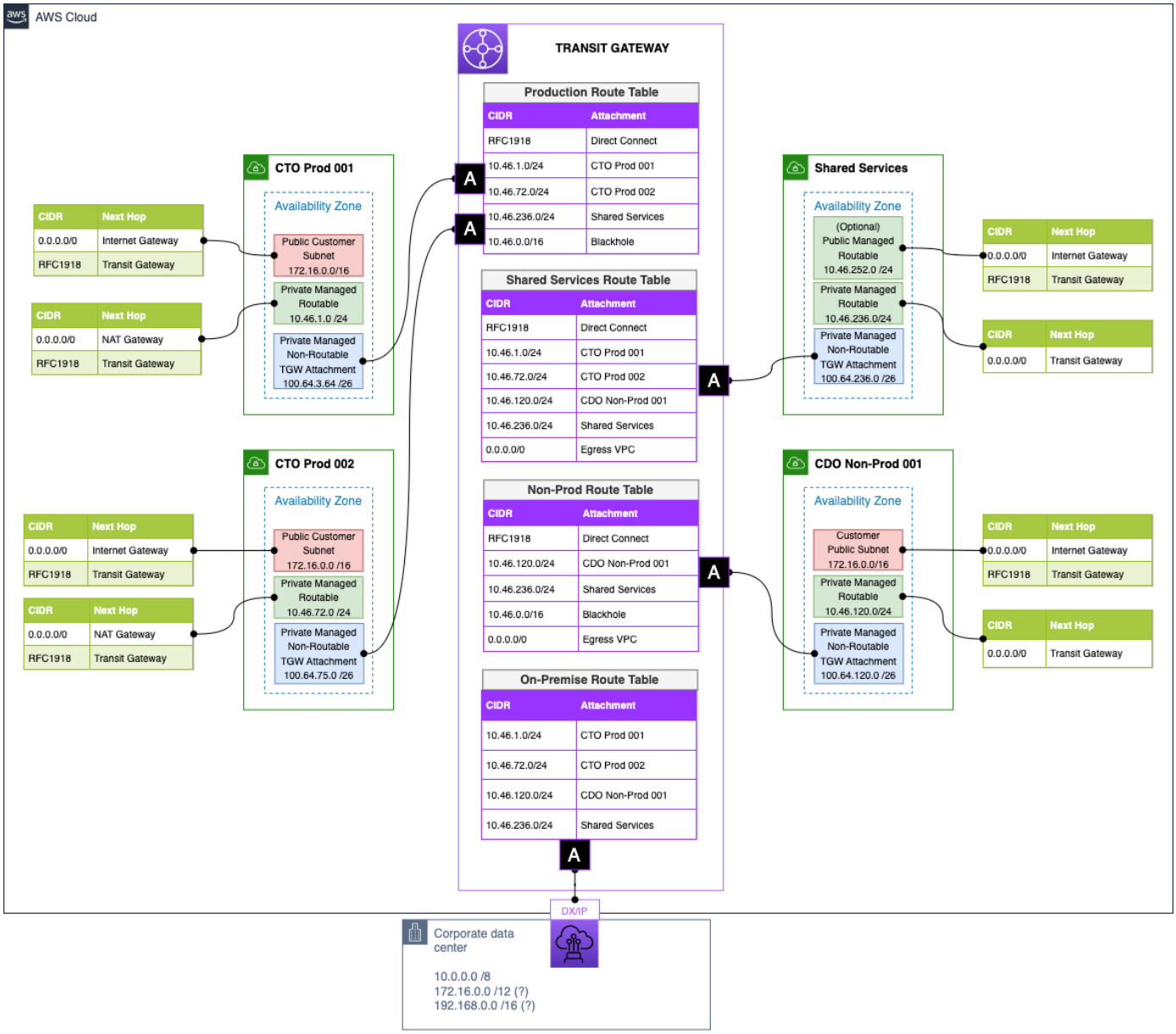


FIGURE 10 - DETAILED ROUTING EXAMPLE

The two accounts are associated with the Production Routing Table in the TGW. The VPC routes for both accounts are propagated or statically assigned to the production routing table. Routing between both accounts is possible via TGW Production Routing table. Traffic is subject to any local security groups and NACLs configured. If the routes are excluded, they cannot communicate each other. The other routes added to the production Routing table allow us to reach the shared services network and the DX Routing table, which is the door to go on-premise.

For the remaining OU's/accounts we will follow a similar procedure, associating the routes to a dedicated routed table, and then controlling the connectivity between them via propagations/static routes.

- Production Routing Table can reach Shared Services and DX.
- Non-Production Routing Table can reach Shared Services and DX.
- No connectivity between Production and Non-Production.
- Diagram shows IGW on Production network. This is not a common practice for production. This scenario is better done via an Egress account with some Firewalling capability at Account level.
- Diagram shows NAT GW as well on Production network. Although NAT is sometime used for package upgrades it's work to consider a safe alternative and use CodeArtifact service for doing that. Allow to proxy securely public repositories for package download, while act a central repository for internal packages as well.

Decision	VMO2 will implement IGW and NAT on the Production network on the initial phase.

To understand the overall routing capabilities is good to see what the model proposed allow us to do. This will explain also the reason behind this approach, allowing the organic growing of the landing zone without any further changes. Figure below is a generic figure that we will transpose to the specific VMO2 case in the next section.

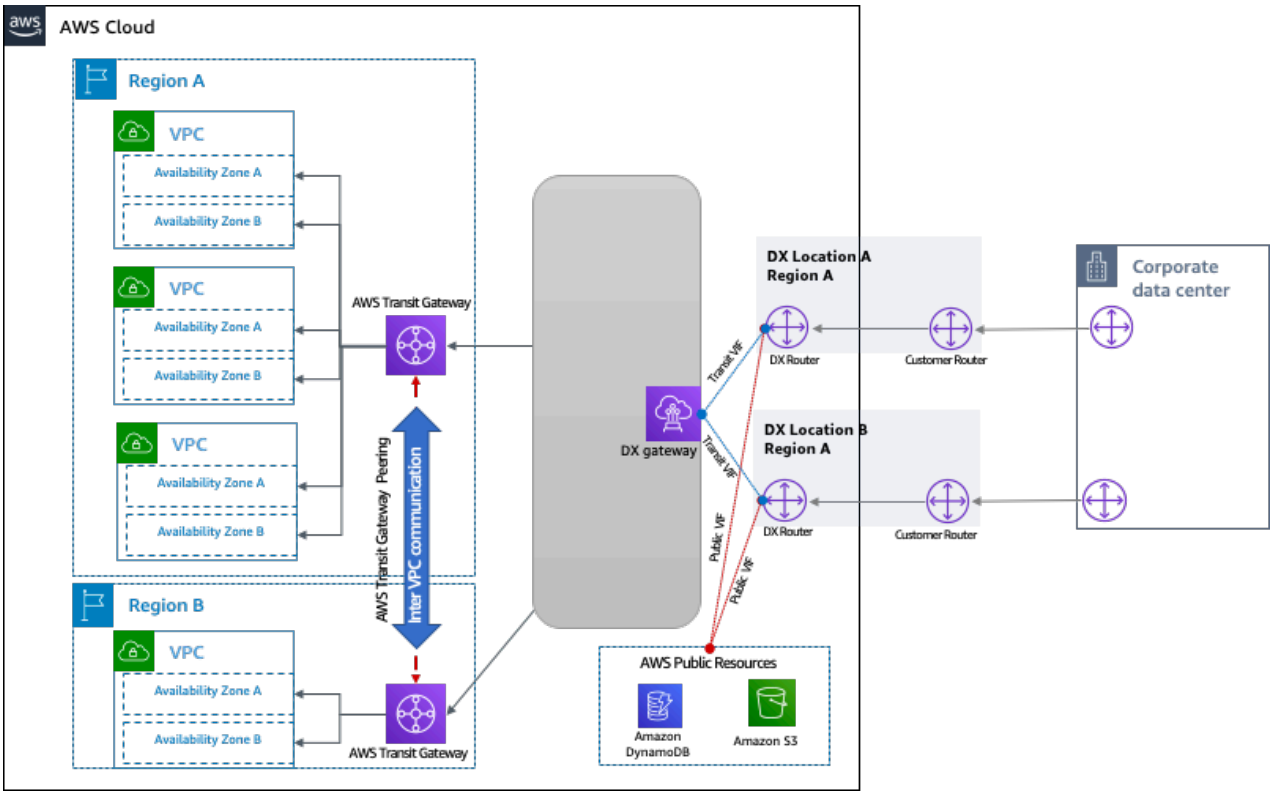


FIGURE 11- GENERIC ON-PREMISE CONNECTIVITY VIA DXGW/TGW

This model is constructed of the following:

- Multi AWS Regions (applicable to single region as well)
- Dual Direct Connect connections to independent DX locations
- Single on-premises data center with dual connections to AWS
- AWS DXGW with AWS Transit Gateway
- High scale of VPCs per Region

Note that the DX location allow via Public VIF to reach the AWS Public Resources (on AWS) privately from On-Premise, without reach the internet. This represents and additional level of security that you have via configuration without additional service activation.

From the routing perspective, when we create the association between the TGW and the DXGW, we specify the "Allowed Prefixes" which the Direct Connect gateway advertises to the on-premises data center. Those prefixes are learnt by the Peer Router on the on-Premise data center, and allow to route from the On-prem the Managed Routable

networks in our cloud area. Here we will apply route summarization and just 10.46.0.0/16 is shared back. Note this is different from the Routing entries on the DX table used on the TGW, where we will keep separate entries to point to the specific subnets supported. By security the remaining subnets not used must be blackholed. The reverse process is also true, when the router data-center share networks via BGP with DXGW, passed back to the TGW. The ones that need to be reachable on-premise from the Cloud, must be advertised.

3.7 DX Routing

The previous schema link

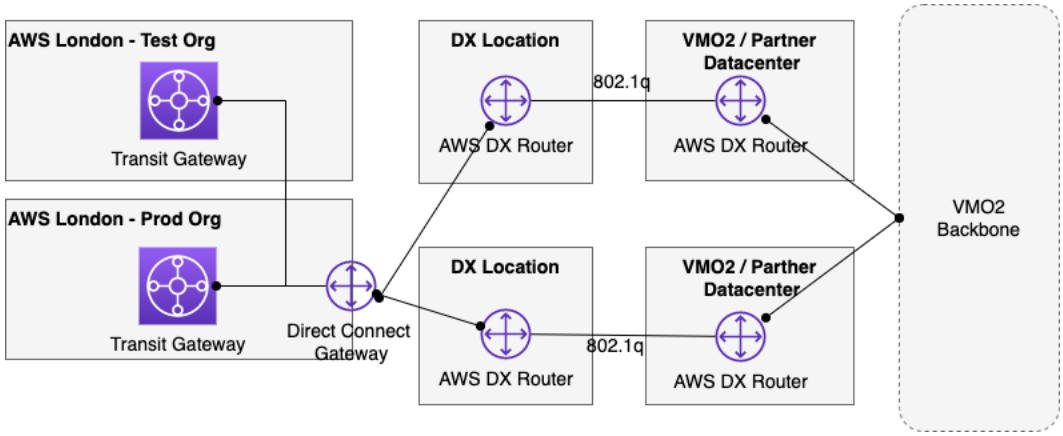


FIGURE 12ON-PREMISE ROUTING DIRECT CONNECT

Routing into/from production and non-production accounts towards VMO2 backbone will be done via DX Connections. A large proportion of the VPCs defined in the AWS VMO2 Cloud will not be routable outside the AWS Cloud. Any source address allowed will be forwarded to the TGW that will have a route point back to DX for all the routes that are being advertised to the AWS cloud.

Traffic initiated on the AWS Cloud will arrive to the DX Routing table, where it has its association. We propagate the VPCs (we can decide which one) into their corresponding routing tables, CxO production to Production routing table and CxO non production to Non-Production Routing table. For the traffic out, the Routing table on which the account is associated need to have a route to the DX attachment. This separation will allow later at any time include an inter-VPC inspection service if it's required.

3.8 Shared Services / Platform OU

This OU will host accounts responsible to implement common services for the organization. Under this OU we are considering the following accounts

- Network - it holds DX connect and TGW.
- DNS - it hosts internal DNS Zones for **aws.private** domain
- Monitoring/Logging - host 3rd party or open-source logging, monitoring and tracing tools
- Common Services - AWS SSM, Base AMIs to be shared with rest of the accounts, Image Builder service, Central ECR repository, etc.
- Egress (not included initially) - it holds the IGW and NAT-GW capability for the company as a central resource.
- Ingress (not included initially) - as per feedback received is not expected to have this requirement. It's recommended to explicitly mention if it's finally not considered or if it be considered in the future and how, centralized or distributed.

Each of those services can be grouped or keep separated from the routing perspective to implement the restrictions needed on the connectivity. Figure below explains how the division has been considered (figure is not exhaustive). There is always the possibility to include the shared Egress and avoid to work locally with internet and NAT GW access at account level. Also note how the Sandbox Account has no connectivity with TGW, remaining isolated from networking perspective from any other account.

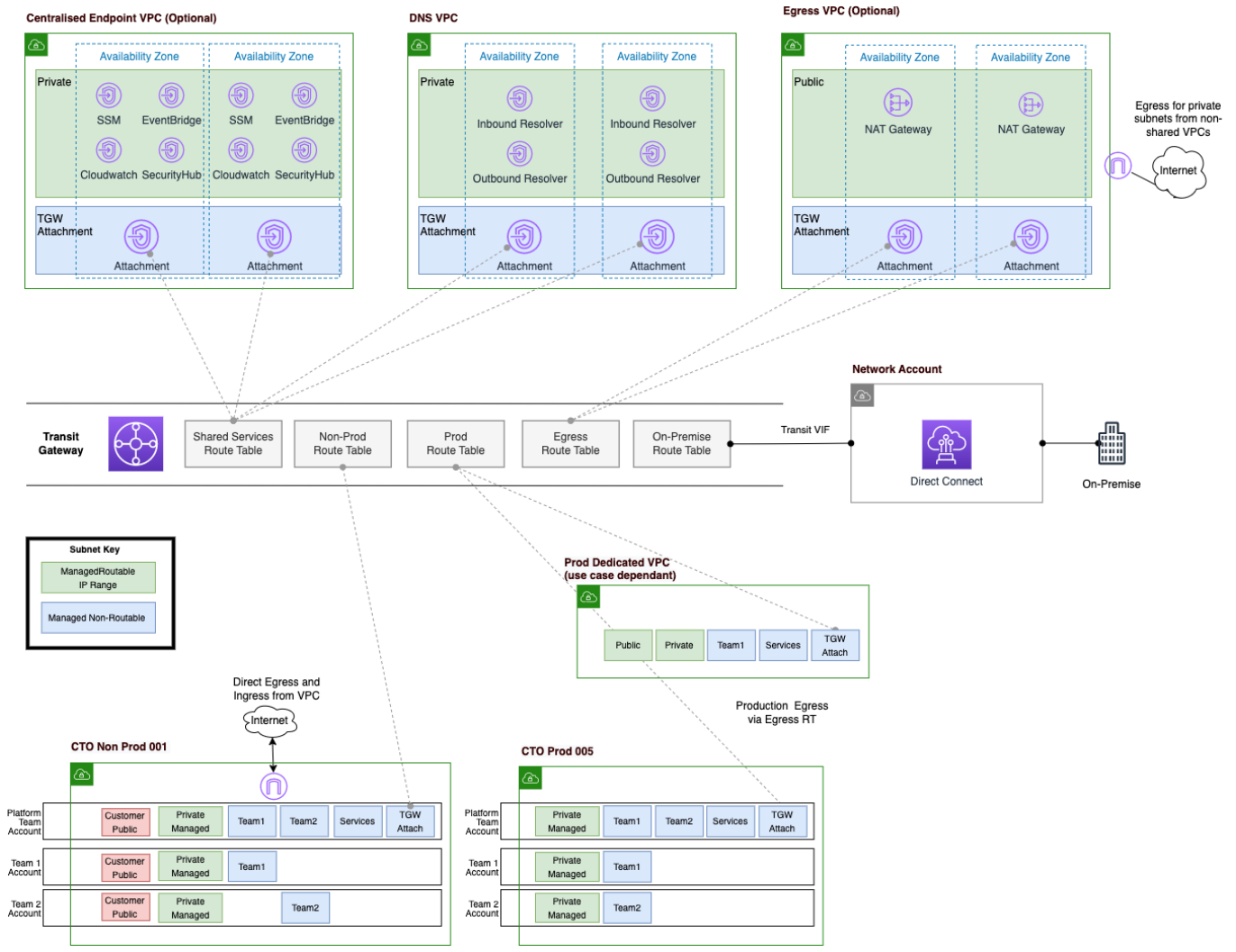


FIGURE 13 - COMPLETE NETWORKING SCHEMA

The public subnet is premise-routable. If we want to avoid this path to happen; On-premise→AWS→Internet, we still can summarize the routes via BGP in the same way and then blackhole the traffic in the AWS side. Since we will not advertise the default route via BGP, this path must not be visible on the on-premise side. If we still get traffic with internet destination via subnet in AWS we can blackhole it on the On-Premise Routing Table on the TGW.

The Shared Services (Platform) OU hosts tools and systems used by the platform operator to manage the platform and resources on it, as such they will have access to all zones and VPC's from a routing perspective but will still be subject to in VPC controls such as NACL's and Security Groups. This means that the services deployed there need to reachable across production and non-production accounts.

3.9 Internet Ingress and Egress Service

3.9.1 Ingress Service

Services which are exposed to the internet will be made available directly ingress Account/VPC, typically these services are terminated on an Application Load Balancer (ALB) or Network Load Balancer (NLB). The ALB and NLB are able to terminate the SSL connection for inbound connections and forward traffic to the services within the VPC. The ALB and NLB are automatically assign an A Record on creation using an AWS subdomain, this can be used as a CNAME on the customer facing DNS name.

To support ingress traffic each public facing VPC will be assigned a public subnet which has a default route to an Internet Gateway (IGW) provisioned within the account. Only one IGW needs to be provisioned per VPC as it is inherently resilient across availability zones.

The service can be implemented distributed or restricted to one specific account. Based on the scenarios we can consider no Ingress, central Ingress, Distributed Ingress, Sandbox Ingress-only.

Decision	VMO2 will not implement Ingress on the initial phase.

3.9.2 Egress Service

Egress connectivity is needed for services which need access to the internet for private subnets. Private hosts may need access to services such as external repositories, APIs, software downloads and patches; this can be provided by using a NAT Gateway in a centralized VPC.

The NAT Gateway provides a Port Address Translation (PAT) for hosts to the internet and is deployed in the public subnet in the Egress VPC. Within the spoke accounts, traffic from private subnets has a default route to the Transit Gateway which routes traffic to the Egress VPC. The NAT Gateway is mapped to an availability zone, for a resilient setup a NAT Gateway is required per availability zone used in the VPC.

Decision	VMO2 will not implement Egress on the initial phase.

3.10 Network Controls

3.10.1NACLs

The goal is to support a flexible delivery model while augmenting the centralized network filtering, egress and ingress patterns. Network Access Control List (NACL) are stateless firewall rules that are applied at a subnet level and have a limitation of 20 rules per NACL. It is recommended that NACL only be applied in a limited manner, wrapping subnets that's have a significant role to play, namely the subnets hosting TGW interfaces or security appliances if deployed and any subnet that is routable to DX. Additionally, NACL's should be managed and automated by the Platform Team (Network/Security) not the application owners.

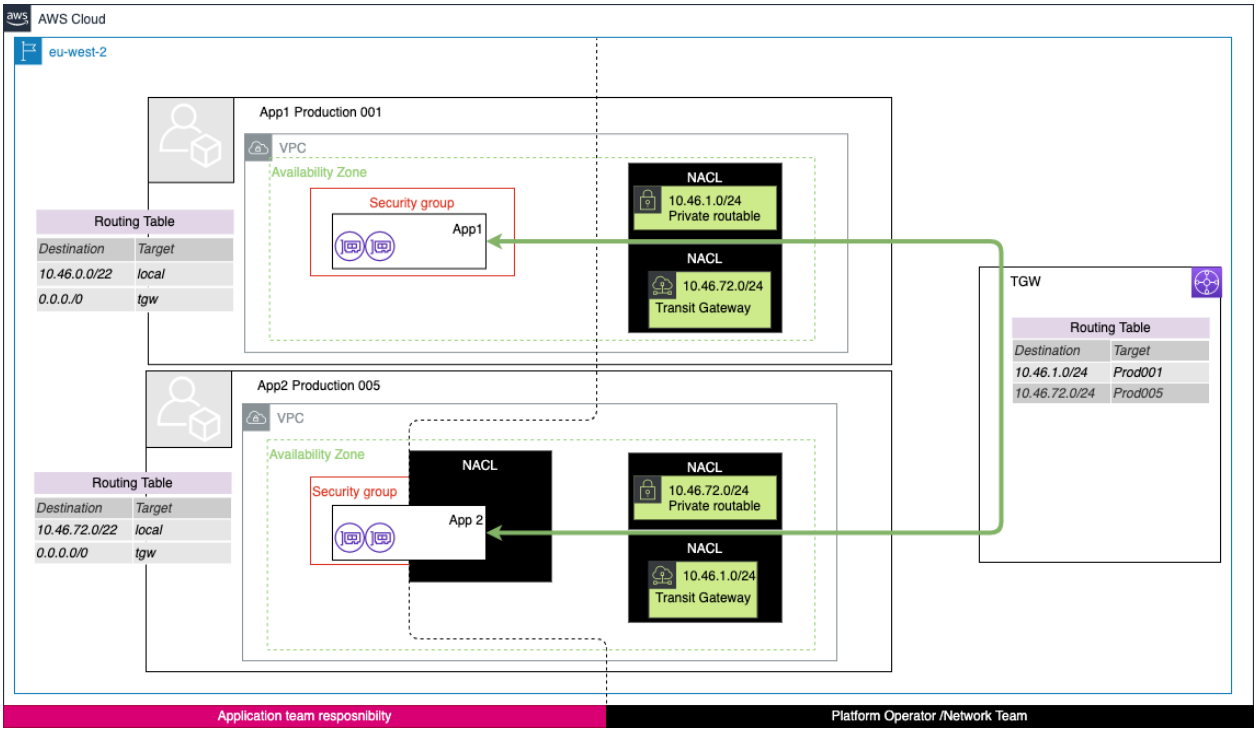


FIGURE 14- NACLS EXAMPLE

3.10.2Security Groups

Security groups (SG) are stateful firewall rules that allow access to and from Elastic Network Interfaces (ENI). SG are more flexible than NACL's and can reference other security groups (although there are some limitations).

As SG's are deployed around network interfaces, they will tend to be application/service specific. Application teams retain responsibility for configuring and managing these rules but guardrails are put in place to detect and alert/remediate when application teams include ingress or egress rules with an allow all (0.0.0.0/0) rule. This includes the default security group.

Decision	Application teams retain responsibility for configuring and managing these rules when they got the capability to do it.

3.10.3Traffic Mirroring

AWS traffic mirroring allows traffic from a VPC or ENI (source) to be encapsulates in a VXLAN header and forwarded across the VPC/TGW network to a target which can be another ENI or network load-balancer. While the mirroring traffic is forwarded across the TGW/VPC networks the ability to setup traffic is controlled through IAM.

Basic setup implies an appliance getting the traffic mirrored (target) from one or many sources. Example below, shows the target D (Appliance) getting traffic mirrored from Source A and B.

FIGURE 15 - TRAFFIC MIRRORING / ACCOUNT LEVEL

If you choose to enable traffic mirroring on Amazon EC2 Instance elastic network interfaces (ENIs), ENI owner pays hourly for each ENI that is enabled with traffic mirroring. But if your intention is to scale this up, is recommended that IAM permissions to perform/setup traffic mirroring along with a Gateway Load Balancer target be limited to the Audit Account where a target Gateway Load Balancer will be used to scale traffic.

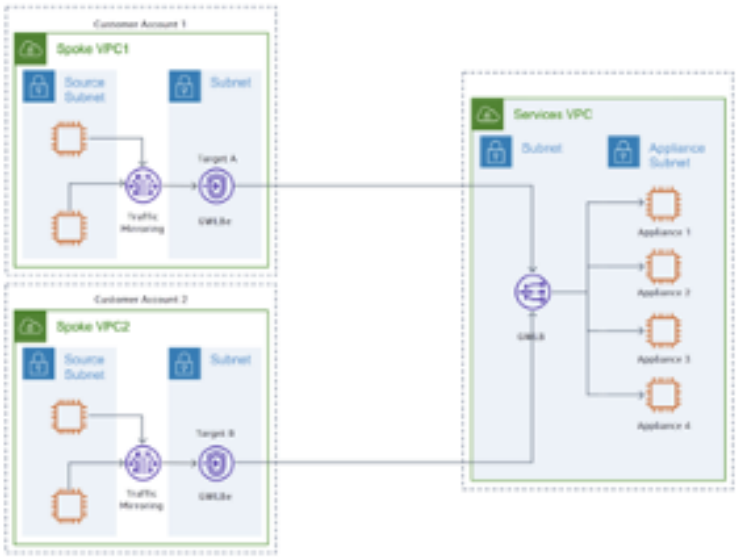


FIGURE 16- TRAFFIC MIRRORING AS CENTRAL SERVICE

The Gateway Load Balancer (GWLB) and Gateway Load Balancer endpoint (GWLBe) permit securely send mirror traffic across VPC and accounts. The GWLBe is a VPC endpoint that provides private connectivity between VPC with the mirror sources and the monitoring appliances deployed behind the GWLB. The GWLB is deployed in a centralized Service VPC with multiple appliances as targets.

3.10.3.1 Traffic Mirroring – pricing example simple case

You enable traffic mirroring sessions on five ENIs in your Amazon VPC in the EU West (London). Traffic mirroring sessions were active for 30 days, 24 hours a day. You will be charged on an hourly basis, for each hour the traffic mirroring sessions were active on ENIs for your Region, the hourly rate is $0.018.

5 sessions (ENI) x 30 days x 24 hr/day x $0.018 per session-hr = $64,8

If account A enables traffic mirroring on an ENI owned by Account B, Account B will be charged for usage.

Decision	Traffic mirroring will not be considered on the VMO2 MVP setup.

3.11 Hybrid DNS

Hybrid DNS refers to the ability to resolve domain name resolution for both AWS and on-premise in both directions, from AWS to on-premise and vice versa. In the diagram below route53 resolves are deployed in a shared services DNS account/VPC and act as the inbound (on-premise to AWS) and outbound (AWS to on-premise) DNS resolvers. This is only relevant for OU's that have on-premise connectivity needs.

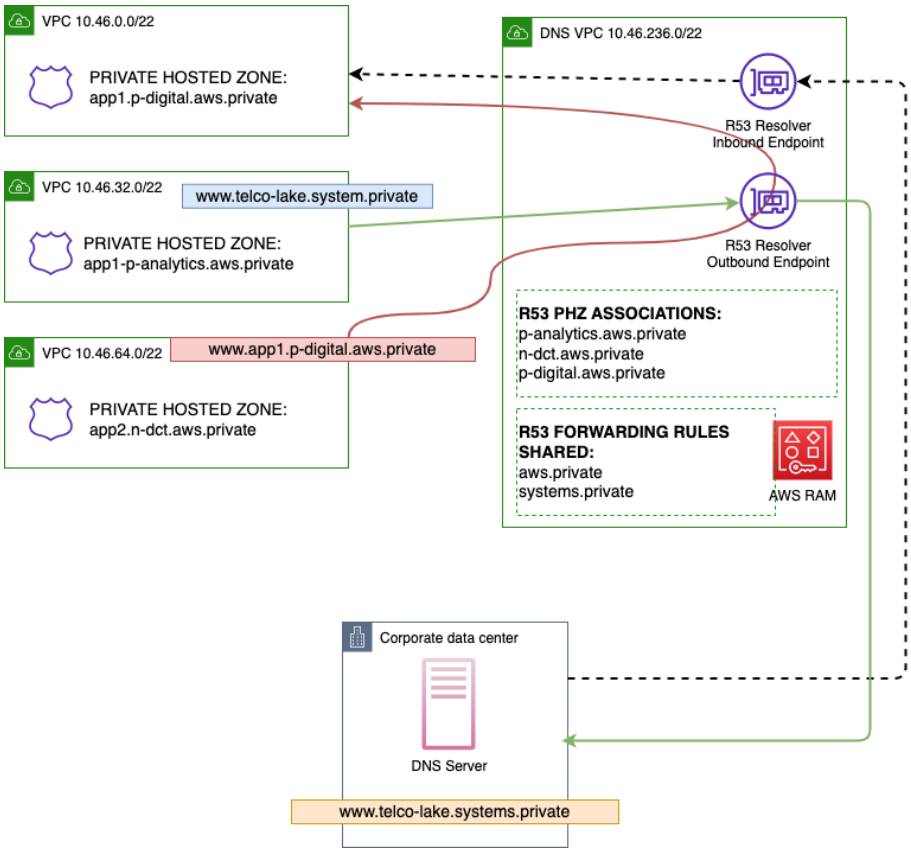


FIGURE 17 - HYBRID DNS / INTEGRATION WITH ON-PREMISE DNS

VPC have a native DNS server built-in, the **.2** resolvers (discussed below), by default this supports internet name resolution and also local VPC name resolution. It's possible to have one app resolving domains in more than one subdomain.

The Platform DNS server does not work as a traditional DNS server and will resolve local VPC DNS names and any route53 private zones which are attached to the VPC, in the example the two zones that are attached to the DNS VPC are *xxx.aws.private*. Each PHZ (private hosted zone) supports standard DNS records such as CNAME, MX etc. as well as AWS alias records that can reference dynamic AWS resources such as load balancers etc. If a domain and record match the request the record value is returned to the DNS client. All traffic towards on-premise will have the domain *xxx.systems.private*

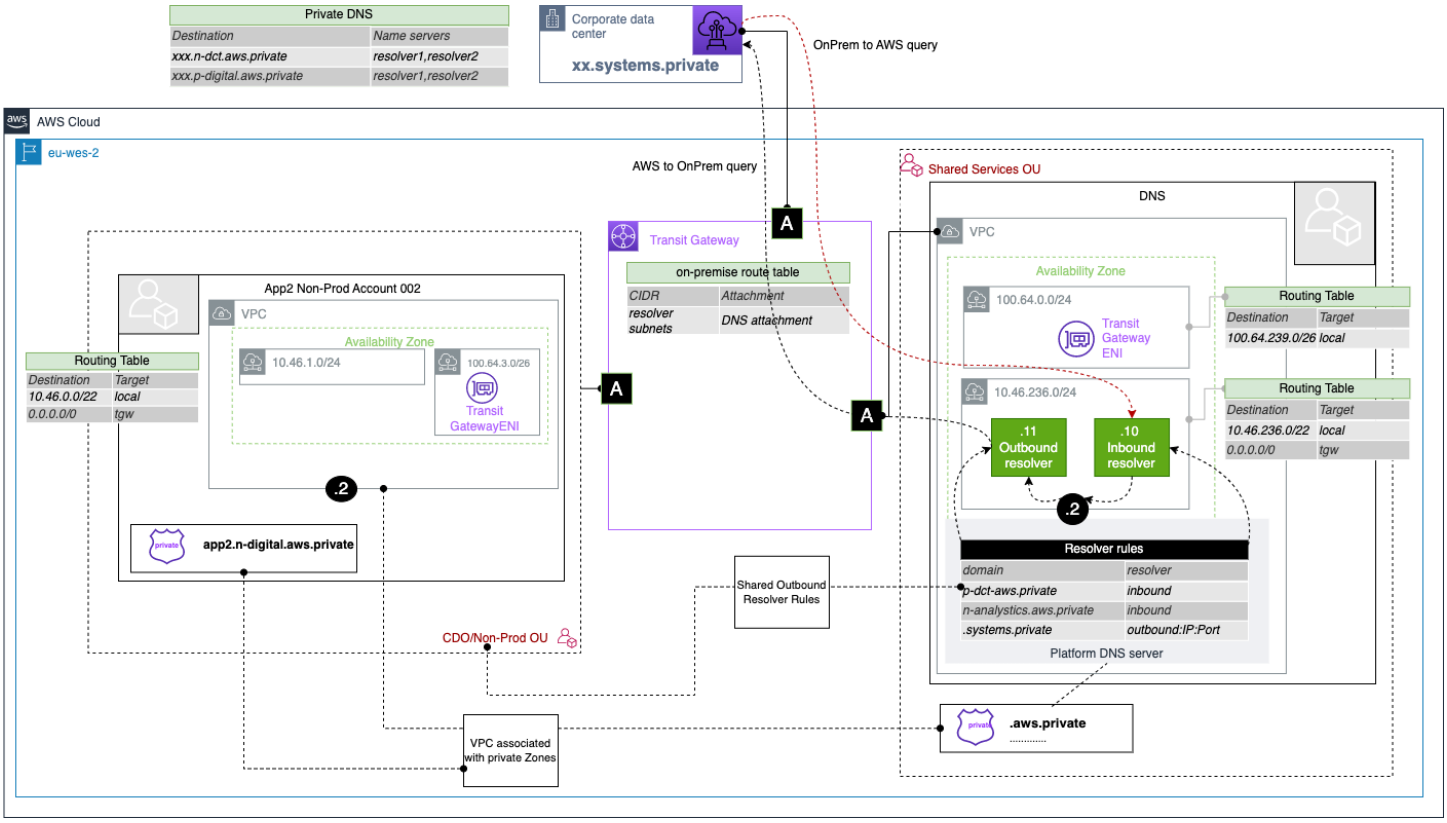


FIGURE 18-DETAILED DNS RESOLUTION

Every VPC that that needs to resolve a private hosted zone must be associated with that zone, minimizing the number of private zones will simplify this process. All connectivity between the VPC and DNS servers is done privately through the AWS control plane, not across the TGW.

On-Prem need to have a resolver on its DNS pointing to the Inbound Resolver in the PHZ at R53. So every time and On-Prem DNS query has as destination a **aws.private** domain, the On-Prem DNS will forward the query to the Inbound Resolver on R53 PHZ, that return the response for that query to the DNS On-prem and then served back to the originator of the query.

3.12 Centralized Endpoints

A VPC endpoint allows you to privately connect your VPC to supported AWS services without requiring an internet gateway or a NAT device or VPN connection, avoiding VPC exposure to the outside world. Traffic between your VPC and other services does not leave the AWS network backbone. They are horizontally scaled, redundant, and highly available VPC components.

Endpoints can be either interface endpoints (powered by AWS *PrivateLink*) or Gateway Endpoints (S3 and DynamoDB). There is no charge for using GW endpoints, but there is charge for using Interface endpoints (per hour + processing costs), for that reason our recommendation is to deploy them centralized.

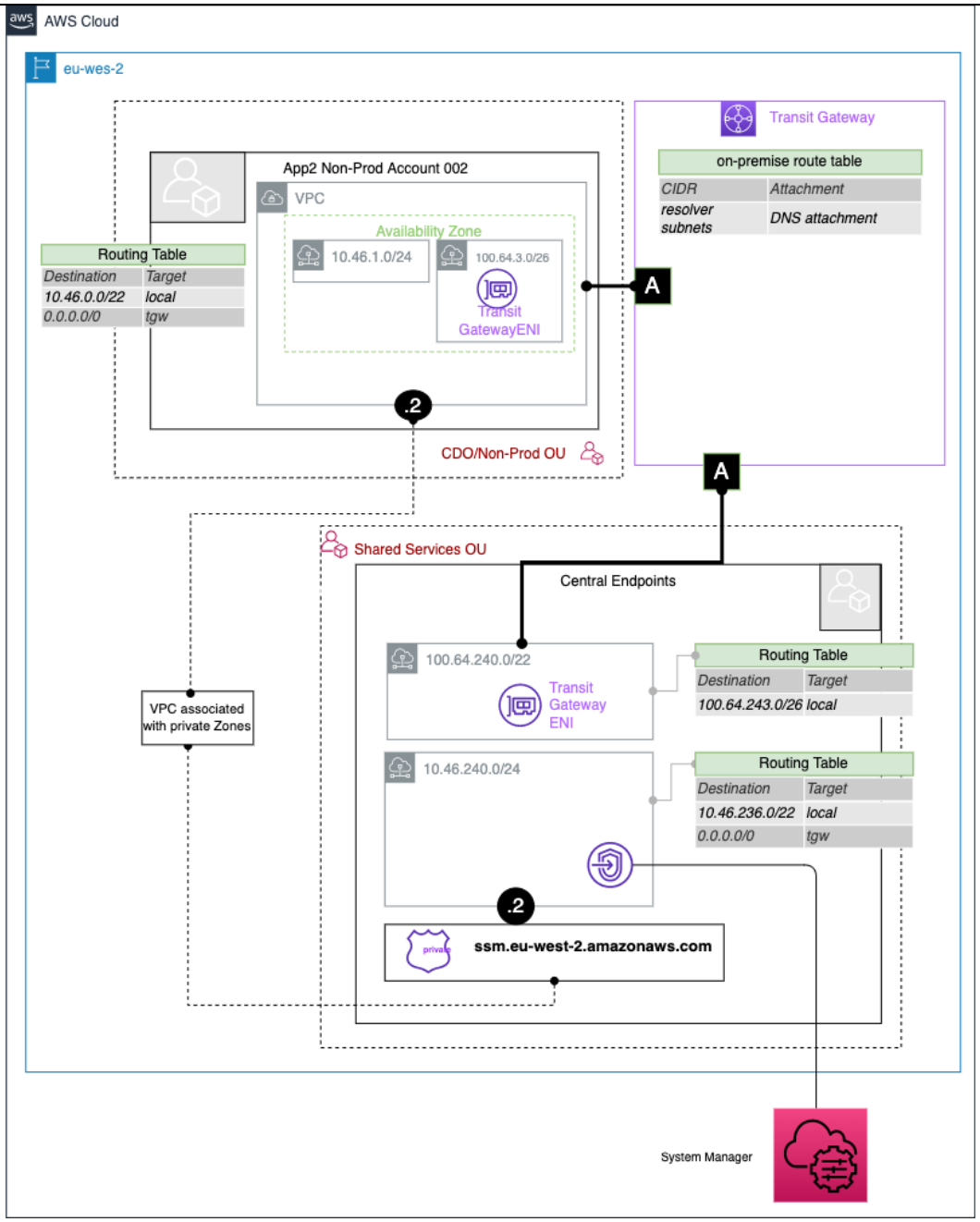


FIGURE 19-CENTRALIZED ENDPOINT DETAILS

When you create a VPC endpoint to an AWS service, you can enable Private DNS. When enabled, the setting creates an AWS managed Route 53 private hosted zone (PHZ) which enables the resolution of public AWS service endpoint to the private IP of the interface endpoint.

The managed PHZ only works within the VPC with the interface endpoint. To overcome this, we will disable the option that automatically creates the private DNS when an interface endpoint is created and manually create a public hosted zone (defined as complete FQDN) that points to the endpoint which can be shared with other VPC's.

Decision	VMO2 MVP setup will align with the recommendation if there is such requirement in the future.

4 Security

4.1 Identity and Access Management

AWS Identity and Access Management (IAM) is a service that provides robust security controls to AWS resources within your AWS environment. The identity and access management is based on the following design tenets:

- Limit access to production environments.
- Simplify identity and group provisioning and management by using a central identity provider
- Use strong sign-in mechanisms such as Multi-factor Authentication
- Leverage User Groups and attributes to scale
- Ensure separation of duties through a set of pre-defined groups that align with VMO2 roles
- Grant least privilege access
- Establish emergency access process

In order to meet these goals, the following technical decisions have been made that influence the design shown below:

- Identity management should be centralized.
- All users and accounts should be subject to multi-factor authentication.
- A combination of granular groups and strict IAM roles should ensure a strong separation of duties.
- No permanent user access should be granted to production, access be to systems (CI/CD, monitoring etc) only.
- Just in Time (JIT) access can be granted for user operational access to production environments.
- Advanced IAM capabilities such as Permission Boundaries and Attribute-based access control (ABAC) should be used to restrict access between teams while supporting developer agility goals.
- Alignment between OU controls (SCP's, Tag Policies, resource sharing, and config rules) and account controls (IAM and Network) is required to provide the required separation and permission boundaries.

4.2 Identity Federation and SSO (Single-Sign On)

Currently VMO2 leverage Okta and AWS SSO to federate users and map users/groups to AWS Account IAM permissions. No changes are needed for this general architect in terms of the way SSO synchronizes users and groups, however the permissions sets, associated account roles and account attachments are recommended to change to fulfill the design tenants listed above. The following diagram illustrates the scope of the design changes.

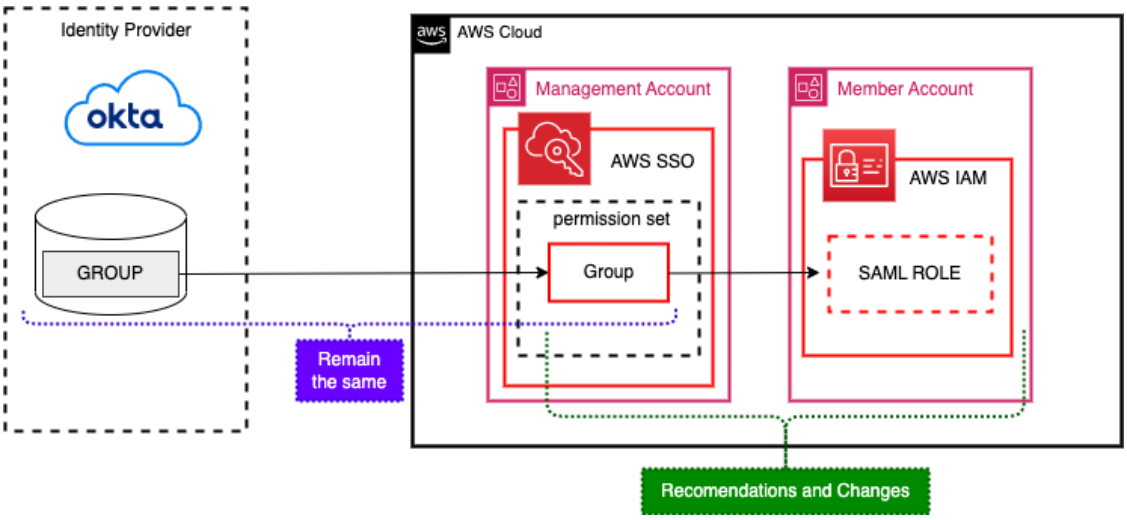


FIGURE 20 - IDENTITY FEDERATION AND AWS SSO

4.3 Authorization Model

The current VMO2 authorization model grants Administrator, Power User etc. access to production and non-production accounts for a mix of roles and users. The recommended model is to align job roles, account types/OU's and a least privilege model.

This will involve

- standardizing access through clear and auditable IAM groups
- using *Just in Time* IAM policies for ad-hoc production user access
- mixing Role Based and Attribute Based Access Control Policies to minimize role and permission sets
- Use of permission boundaries to establish clear guardrails and delegate IAM role creation.

- Standard roles for cross account activities such as Break Glass and CI/CD

The following diagram illustrates the IAM building blocks.

- A policy, stores a given set of permissions (PERMIT/DENY).
- A Role, maps a principal to a policy and defines what can be done by a specific element
    - RBAC roles explicitly map a principal to a policy and optionally a specific set of resources
    - ABAC roles map a principal (based on an attribute) to a set of resources (again based on a set of attributes/tags) to a policy
    - permission boundary, limited what a role can do, irrespective of what the role is configured to do

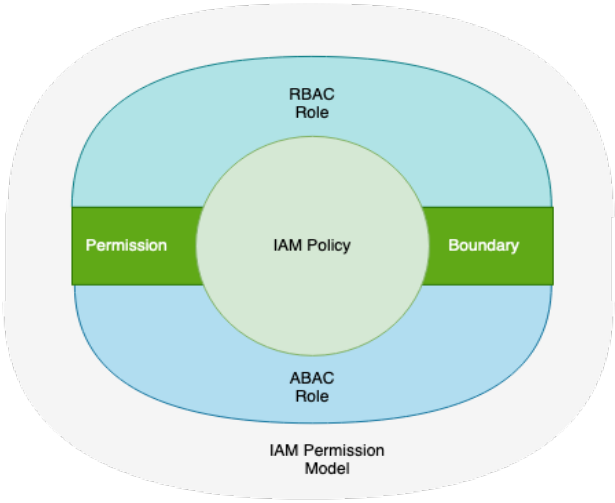


FIGURE 21- IAM POLICY MODEL

4.4 Okta group naming convention

All groups must to be mapped with Okta groups/security groups . Okta will use the following naming convention:
Platform Groups:

- AWS-Cloud-{GROUP-NAME}
    - GROUP-NAME = Admins, Engineers, Users, Security
- AWS-Billing-Users
- AWS-Platform-Approvers

Member Groups:

- AWS-{DIVISION}-{PRODUCT}-{GROUP- NAME}
    - DIVISION = cdo, cto, cio.
    - PRODUCT = Freeform, it will be name of application/platform (e.g. connect).
    - GROUP-NAME = Admins, Developers, Users

4.5 Groups / Permission-Set Mapping

The following table defines the standard groups that are recommended and the mapping initially considered by VMO2.

OKTA Group Name	Job Function	AWS IAM Group	AWS IAM Permission Set	AWS IAM Policies	Accounts
AWS-Cloud-Admins	Platform Admin	AWS-Cloud-Admins	AWSAdministratorAccess	AdministratorAccess	Management
AWS-Cloud-Engineers	Platform Engineer	AWS-Cloud-Engineers	AWSCloudEngineers AWSSecurityAuditors	NetworkAdministrator SystemsAdminstrator SupportUser	SharedServices
AWS-Billing-Users	FinOps Engineer	AWS-Billing-Users	AWSCloudBilling	Billing	Management
AWS-Cloud-Security	Security Engineer	AWS-Cloud-Security	AWSLogArchiveAdmins AWSSecurityAuditors	SecurityAudit ViewOnlyAccess	Log-Archive Audit
AWS-ReadOnly-Users	Viewer	AWS-ReadOnly-Users	AWSReadOnlyAccess	ReadOnlyAccess	ALL

Members Accounts Groups

OKTA Group Name	Job Function	AWS IAM Group	AWS IAM Policy	Accounts
account-email@vmo2.co.uk	Account Owner	Root	Administrator	
AWS-<Division>-<Product>-Admins	Product Admins	AWS-<Division>-<Product>-Admins	ViewOnlyAccess SupportUser Admin Role (?) Extra Role via IaC	Related Accounts to Product with Division
AWS-<Division>-<Product>-Developers	Product Developers	AWS-<Division>-<Product>-Developers	ViewOnlyaAccess Extra Role via IaC	Related Accounts to Product with Division
AWS-<Division>-<Product>-Users	Product Users	AWS-<Division>-<Product>-Users	ViewOnlyaAccess Extra Role via IaC	Related Accounts to Product with Division

Additional possible permission sets to be considered:

Permission-Set	Description/Permission Set	Functional Group
AWSLogArchiveViewers	Read-only access to the log archive account only	AWS Platform Engineering AWS-<Division>-<Product>-*
AWSAuditAccountAdmins	Full access administrator rights to the audit account only	AWS Platform Engineering
ProductAdmin	Power user to AWS Services but limited to AWS accounts (RBAC) or resources tagged with appropriate Key/Value (ABAC. Please note this role would be extended from the AWS managed roles to contain DENY statements for **S3,DynamoDB (data restrictions) that contain senstive data** and could also support the creation of IAM roles with attached permission boundaries.	AWS-<Division>-<Product>-*
ProductDataAdmin	read/write access to all S3 Data stores	AWS-<Division>-<Product>-*
ProductKeyManger	Permission to manage encryption keys (KMS) limited to AWS accounts (RBAC) or resources tagged with appropriate Key/Value (ABAC)	AWS-<Division>-<Product>-Admins
NetworkEngineerRO	Custom Role with Read-only access to other accounts - VPC - CloudWatch	AWS-<Division>-<Product>-Developers
NetworkAdmin	read/write cross account - VPC - EC2 networking - CloudWatch	AWS-<Division>-<Product>-Admins

(*) The asterisk on the functional groups means to all the groups that require the services based in your needs. The table is a suggested assignment.

4.6 Policy Evaluation Logic

The following flow chart provides detail on how any IAM decision is evaluated.

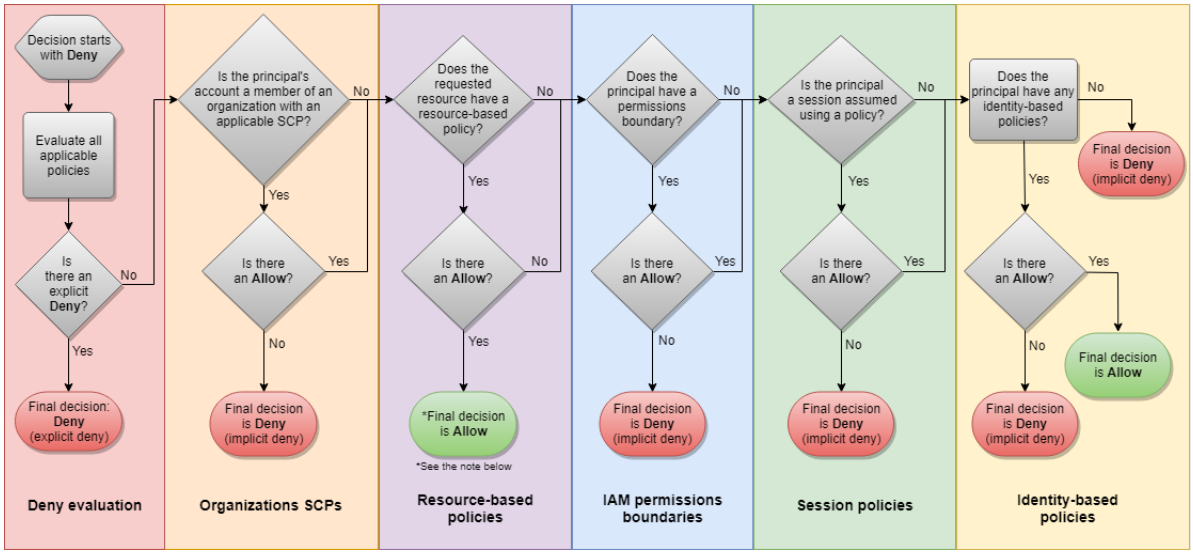


FIGURE 22 - GENERAL POLICY EVALUATION LOGIC

Deny evaluation – By default, all requests are denied. This is called an implicit deny. The AWS enforcement code evaluates all policies within the account that apply to the request. These include AWS Organizations SCPs, resource-based policies, IAM permissions boundaries, role session policies, and identity-based policies. In all those policies, the enforcement code looks for a Deny statement that applies to the request. This is called an explicit deny. If the code finds even one explicit deny that applies, the code returns a final decision of Deny. If there is no explicit deny, the code continues.

Organizations SCPs – Then the code evaluates AWS Organizations service control policies (SCPs) that apply to the request. SCPs apply if the request is made in an account to which the SCP is attached. If the enforcement code does not find any applicable Allow statements in the SCPs, then the request is implicitly denied. The code returns a final decision of Deny. If there is no SCP, or if the SCP allows the requested action, the code continues.

We will see the remaining ones in more detail in the following sections.

4.7 Role Based Access Control (RBAC)

RBAC is the standard way of authorizing access to resources. In this model SSO groups are mapped to permission sets and linked to specific accounts. In the table below the shows the mapping between the VMO2 teams/roles and the different account OU's.

4.7.1 Permission Boundaries

A permissions boundary is an advanced IAM feature that helps your centralized team, such as VMO2 Security Team, to allow application developers to create new IAM roles and policies. In the table above DevOps and Dev teams only have Dev-x access so they cannot create roles or policies. This would need to be a Platform Team member or the CI/CD Pipeline unless permission boundaries were used.

A permissions boundary is designed to restrict permissions on IAM principals, such as roles, such that permissions don't exceed what was originally intended. The permissions boundary uses an AWS or customer managed policy to restrict access, and it's similar to other IAM policies as it has resource, action, and effect statements.

A permissions boundary alone doesn't grant access to anything. Rather, it enforces a boundary that can't be exceeded, even if broader permissions are granted by some other policy attached to the role.

For example:

- The VMO2 Platform/Security team create a policy, the permission boundary that will limit the ability to create EC2 instances in eu-west-2 (London) shown below (this is an example, regional controls like this would work better with Service Control Policies, that can be covered with CfCT/TF IaC). This type of policies can be enforced at OU level then applied to the Account automatically when added to the OU.

```
{"Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "EC2RestrictRegion",
      "Effect": "Allow",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": ["eu-west-2"]
        }
      }
    }
  ]
}
```

- The VMO2 Platform/Security team adds a condition (permission policy) to the developer's SSO permission set (inline policy) to ensure that any new role must have the permissions boundary (created in step 1) attached to it. This is similar to the way the JIT condition works.

  Example of Developer Permission Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ReadRoles",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles",
        "iam:ListPolicies",
        "iam:GetRole",
        "iam:GetPolicy",
        "iam:ListAttachedRolePolicies",
        "iam:ListInstanceProfilesForRole",
        "iam:ListRolePolicies"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowRoleCreationWithAttachedPermissionsBoundary",
      "Effect": "Allow",
      "Action": [
        "iam:CreateRole",
        "iam:AttachRolePolicy"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/EC2RestrictRegion"
        }
      }
    },
    {
      "Sid": "DenyPermissionsBoundaryDeletion",
      "Effect": "Deny",
      "Action": "iam:DeleteRolePermissionsBoundary",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:PermissionsBoundary": "arn:aws:iam:: 123456789012:policy/EC2RestrictRegion"
        }
      }
    },
    {
      "Sid": "DenyPolicyChange",
      "Effect": "Deny",
      "Action": [
        "iam:CreatePolicyVersion",
        "iam:DeletePolicyVersion",
        "iam:DetachRolePolicy",
        "iam:SetDefaultPolicyVersion"
      ],
      "Resource": "arn:aws:iam:: 123456789012:policy/EC2RestrictRegion"
    }
  ]
}
```

The developer creates a new role in their account with accompanying permissions boundary (EC2RestrictRegion) assigned to create EC2 resources (`"ec2:*"`). The role can used to create EC2 instances but only in eu-west-2 due to the (enforced) attached permission boundary.

4.8 Attribute Based Access Controls

Attribute Based Access Control is a model which can leverage the attributes from the VMO2 Okta Directory and provide access when the AWS resources have the same tag. The following list describes the benefits of using ABAC.

- **ABAC requires fewer permission sets –** Because you don't have to create different policies for different job functions, you create fewer permission sets. This reduces your permissions management complexity.
- **Using ABAC, teams can change and grow quickly –** Permissions for new resources are automatically granted based on attributes when resources are appropriately tagged upon creation.
- **Use employee attributes from your corporate directory with ABAC –** You can use existing employee attributes from any identity source configured in AWS SSO to make access control decisions in AWS.
- **Track who is accessing resources –** Security administrators can easily determine the identity of a session by looking at the user attributes in AWS CloudTrail to track user activity in AWS.

In the example below, the *CostCenter* attribute is received from the Okta Directory (`aws:PrincipalTag/CostCenter`) and is matched to the CostCenter tag attached to the EC2 instances (`ec2:ResourceTag/CostCenter`).

As example; if the Cost Center is "***CTO***" then anyone assigned this attribute in their profile would be able to perform DescribeInstances, StartInstances & StopInstances API requests for any EC2 instance tagged with CostCenter=**CTO**

```
{"Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ec2:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"
        }
       }
     }
   ]
}
```

Note that ABAC requires a good tagging strategy and controls such as Tagging Policies, IAM Policies and Config rules to ensure resources are always tagged with the appropriate tags so that IAM controls work correctly.

Decision	VMO2 will conside the usage of ABAC role in a more mature phase, once the Tagging is more mature.

4.9 Just in Time Permission Sets

Development, Platform and DevOps teams/role will not by default have access to production accounts, ia JIT policy can be created with a Condition ***DateGreaterThan & DateLessThan*** that equals the time required for the access by the Platform Team. This can then be associated with a SSO Group/Account when access is needed. Bear in mind user/groups would need 40+ minutes to synchronize with SSO so new users or groups that needed JIT access would ideally be pre-provisioned.

Note that the AWS IAM policy framework will always prefer an explicit deny over an explicit allow so care must be taken that any JIT policy (allow) is not superseded by an explicit deny in an existing group or user policy.

In the example shown below the principal (SSO group) would be able to describe AWS Certificate Manager certificates between the dates 13:06:2022-14:06:2022 (lines 10 and 11).

```
{"Version": "2012-10-17",
   "Statement": {
   "Effect": "Allow",
   "Action": [
    "acm:DescribeCertificate",
    "acm:ListCertificates",
    "acm:GetCertificate",
    "acm:ListTagsForCertificate"
   ],
   "Condition": {
         "DateGreaterThan": {"aws:CurrentTime": "2022-06-13T00:00:00Z"},
         "DateLessThan": { "aws:CurrentTime": "2022-06-14T23:59:59Z"}
    },
    "Resource": "*"
   }
 }
```

4.10 Break Glass Approach

The break glass process provides high-privilege access to AWS accounts and resources, and allows users to access those assets during emergencies where regular administrative accounts (Okta Directory or AWS SSO) are inaccessible. In order to recover from some incidents, it may become necessary to circumvent standard access controls. The following scenario must occur to trigger the break glass process:

- MFA: Multifactor authentication may not be available, or a network/cellular outage renders the MFA unreachable.
- Single Sign-On: Access via federation from the Okta domain is not possible and hence the AWS environment is unreachable.

While federated authentication is being re-established and emergency access is needed to any of the VMO2 AWS accounts, the following break glass process is activated:

- The person activating the process will submit a request to the platform team.
- The breakglass account has a breakglass group that allows its users to assume a breakglass role in all accounts in the Landing Zone. The platform team has IAM users that they can add to the breakglass group and create breakglass IAM users with privileged access in the target accounts of the requestor. The platform team then hands over the credentials to the requestor.
- The passwords of the break glass users need to comply with the VMO2 password policy and use MFA. The platform team revokes the access from the requestor (by user deletion) when the incident is resolved.

The following roles will be created in the member/child accounts independently of AWS SSO. The usage of Cloudformation StackSet allow to deploy in all the accounts this role defining natively in Cloudformation or Terraform.

Role	Use Case	Permissions	Trusted Principals
VMO2-role-breakglass	Role to be assumed from the break glass account	AdministratorAccess *(note this is a placeholder, ideally this will be reduced to minimal set of permissions required by the breakglass role)*	Break Glass account

All AWS API actions will be logged, using CloudTrail, to a central bucket and will contain a ***userIdentity*** key (see below) which will reflect the break class roles/identity.


```
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789:Bob",
    "arn": "arn:aws:sts::123456789:assumed-role/123456789/Bob",
    "accountId": "123456789",
    "accessKeyId": "123456789",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2019-08-05T07:15:25Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789",
        "arn": "arn:aws:iam::123456789:role/123456789",
        "accountId": "123456789",
        "userName": "Bob"
      }
    }
```


This data can be queried using AWS Athena (see example) for forensic examination or integrated with a Lambda to perform real-time alerting/remediation when a breakglass IAM identity is used.

4.11 Controls (a.k.a GuardRails[1])

AWS Control Tower implements **preventive**, **detective**, and **proactive** controls that help you govern your resources and monitor compliance across groups of AWS accounts. A control applies to an entire organizational unit (OU), and every AWS account within the OU is affected by the control. Therefore, when users perform work in any AWS account in your landing zone, they're always subject to the controls that are governing their account's OU.The Control Tower relevant controls from the infrastructure perspective

4.11.1 Control/GuardRail behavior

- **Preventive** (Mandatory) – A preventive will ensure that accounts maintain compliance, because it disallows actions that lead to policy violations. The status of a preventive guardrail will be either **enforced** or **not enabled**.

- **Detective** (Optional) – A detective guardrail will detect non compliance of resources within the accounts, such as policy violations, and will provide an alert through the dashboard. The status of a detective guardrail will either be **clear**, **in violation**, or **not enabled**.

- **Proactive** (Optional) - A proactive control scans your resources before they are provisioned, and makes sure that the resources are compliant with that control. Resources that are not compliant will not be provisioned. Proactive controls are implemented by means of AWS CloudFormation hooks, and they apply to resources that would be provisioned by AWS CloudFormation. For that reason, on the remaining of this section we will refer to the Preventive and Detective ones, since those all applicable to the generic Infrastructure and the use of other IaC Tools.

[1] AWS is transitioning the terminology to align better with industry usage and with other AWS services. During this time, you may see the previous term, *guardrail*, as well as the new term, *control*, in our documentation, console, blogs, and videos. These terms are synonymous for our purposes.

4.11.2 Implementation of guardrail behavior

- The preventive/Mandatory guardrails are implemented using Service Control Policies (SCPs), which are part of AWS Organizations.
- The detective/Optional guardrails are implemented using AWS Config rules and AWS Lambda functions.
- The root user and any IAM administrators in the management account can perform work that guardrails would otherwise deny. This exception will prevent the management account from entering into an unusable state. All actions taken within the management account continue to be tracked in the logs contained within the log archive account, for purposes of accountability and auditing

4.11.3Guardrails/Control Guidance in Control Tower

- Mandatory controls (Preventive) are owned by AWS Control Tower, and they apply to every OU on your landing zone. These controls are applied by default when you set up your landing zone, and they can't be deactivated. You cannot turn them off for any OU.

- Optional Controls (Detective) - *Detective controls* are security controls that are designed to detect, log, and alert after an event has occurred. Detective controls are a foundational part of governance frameworks. These guardrails are a second line of defense, notifying you of security issues that bypassed the preventative controls. Under this category we want. Optional controls in AWS Control Tower are applied at the OU level.

    - Strongly recommended controls are based on best practices for well-architected multi-account environments. These guardrails are not enabled by default and they can be enabled/disabled through the AWS Control Tower console, the control APIs, CfCT (Customizations for Control Tower) or other IaC tool.

    - Elective controls (enable you to lock down or track attempts at performing commonly restricted actions in an AWS enterprise environment. These controls are not enabled by default, and can be disabled. Based on SCPs and AWS Config Rules.

        - Data Residency Controls - These elective controls complement your enterprise's data residency posture. Help detect and inhibit the purposeful or accidental creation, sharing, or copying of data, outside of your selected AWS Region or Regions.

    - Security Hub Standard/Additional Config Rules can be deployed by the AWS Security Hub to monitor compliance with standards such as the CIS AWS Foundations Standard. It will help you align with Well-Architected best practices.

- Proactive Controls – are optional controls implemented by AWS CloudFormation Hooks.

Decision	Proactive Controls are out of the scope of MVP initially due to its dependency on the CFN Hooks. Those can be considered in a later stage if needed.

Decision	The selected guardrails and their respective mapping will follow the VMO2 selections accordingly to the table documented on this link: https://galactic.virginmedia.co.uk/confluence/display/PCP/AWS+SCPs+and+Config+Rules

- All the Mandatory GuardRails will be implemented.
- Strongly Recommended and selected Elective Guardrails will be implemented on the Landing Zone at required OU levels based on the matrix identified on the previous link.
- Custom Guardrails, as per the captured requirements that could appear on the implementation phase will be built and deployed as well.

4.12 Infrastructure Security

4.12.1Network Security

- **AWS WAF** - AWS WAF is a web application firewall that monitor web requests that are forwarded to an Amazon CloudFront distribution, an Amazon API Gateway REST API, an Application Load Balancer, or an AWS APPSync GraphQL API.
- **AWS Shield** - AWS provides DDoS protection service, known as AWS Shield.
    - Standard version is enabled by default and it provides protection against the most common network layer (layer 3) and transport layer (layer 4) attacks. AWS Shield Standard does not provide metrics on DDoS events.
    - Advanced mode provides metrics via CloudWatch. AWS Shield Advanced only protects resources that you have specified either in Shield Advanced or through an AWS Firewall Manager Shield Advanced policy. It doesn't automatically protect your resources. It provides additional protections against more sophisticated and larger attacks for your applications running on Amazon EC2, Elastic Load Balancing (ELB), Amazon CloudFront, AWS Global Accelerator, and Route 53.
- **AWS Network Firewall** - AWS Network Firewall is a managed service that can be deployed within VMO2 VPC's and automatically scales with the network traffic. It inspects traffic flows that are routed through it. It can be deployed centrally via inspection account or distributed in each account.
- **AWS Firewall Manager** – Allow multi-account management. With Firewall Manager, you can set up your AWS WAF firewall rules, Shield Advanced protections, Amazon VPC security groups, Network Firewall firewalls, and DNS Firewall rule group associations just once. If you add a new account, it becomes automatically configured.
- **DNS Firewall** – managed integration of Route53 into Firewall manager.

Decision	VMO2 will activate Shield Advanced.

Decision	VMO2 will not deploy Network Firewall Initially.

Decision	The usage of the other services can be postponed until the MVP is ready. A decision on those can be taken later, but it can benefit from a joint and broad implementation using CfCT for SCPs and Security Services, although is not gating other services.

4.12.2 Service Catalog

Many AWS customers are adopting AWS Service Catalog to create and manage catalog of approved IT services for use on AWS. The AWS Service Catalog Hub-and-Spoke model enables organizations to centrally manage IT services they want to distribute to their lines of business (LOBs).

There are numerous benefits to sharing AWS Service Catalog products in AWS Organizations, such as:

- The ability to create multiple hubs of AWS Service Catalog portfolios depending on logical AWS account groups in your company
- The ability auto-share AWS Service Catalog portfolios with newly created AWS accounts in the AWS Organization or Organizational Unit
- A single AWS account to manage sharing of infrastructure services to different parts of the company using AWS Service Catalog
- Product updates, additions, and deletions from the hub reflect across the organization automatically.

At VMO2, AWS Service Catalog portfolios will be created and shared with the Organization from the Management Account[2]. The creation of the service catalog can be programmatically and initially, we will use the Service Catalog Account Factory to enable the account creation programmatically.

4.12.3Encryption at rest

VMO2 will use server-side encryption to protect data at rest.

VMO2 will take a ubiquitous encryption approach to data protection by enabling encryption for all services which natively support KMS encryption. Business justification will have to be provided to be exempt from this rule.

Default encryption will be enabled on S3 buckets and EBS volumes if supported instance types are used.

Decision	AWS KMS will be used as the key management solution. In cases where KMS integration is not natively supported by a service, VMO2 will determine the approach based on the data classification model.

Decision	Secrets will be handled by VMO2 HasiCorp Vault

4.12.4Encryption in transit

VMO2 will enforce the use of HTTPs for all public connections and may utilize AWS ACM to provision and manage TLS certificates for VMO2's public services and applications. Implementing secure key and certificate management, enforce encryption in transit and authenticating network communications align to Well-Architected best practices.

Decision	AWS Session Manager will be used for administrative access to EC2 instances

4.13 Detective Controls

All the mechanisms explained in this section and some other additional ones, can be enforced automatically as IaC using a AWS managed solution named "*Customizations for Control Tower*". This is an implementation pipeline that need to be created (it's automated too) in the management account and allow to automatically align the Control Tower deployment with the AWS SRA (Security Reference Architecture):

4.13.1AWS Security Hub

AWS Security Hub consumes, aggregates, and analyses security findings from various supported AWS and third-party products. Security Hub also generates its own findings as the result of running automated and continuous checks against the compliance rules in the supported security standards. These checks provide a compliance score and identify specific accounts and resources that require attention.

Security Hub supports the below standards:

- **CIS AWS Foundations**, follows the CIS Benchmark for CIS AWS Foundations Benchmark, v.1.2, Level 1/2.
- **AWS Foundational Security Best Practices**, is a standard set of controls that detect when your deployed accounts and resources deviate from security best practices. It continuously evaluates all of the AWS accounts and workloads to quickly identify areas of deviation from best practices. It provides actionable and prescriptive guidance on how to improve and maintain your organization's security posture. The controls include best practices from multiple AWS services. Each control is assigned a category that reflects the security function that it applies to. See Control Categories.

[2] Any account with Service Catalog Administrator capabilities can create product into the Service Catalog. This is associated to the management account since it's created when the AWS Organization Root Account is created. It's a AWS Organizations capability. This doesn't imply that the management account must be used to work on the portfolio and product management on it. Golden images can be included on the Service Catalog as part of a product that need those images, as such enforcing their usage.

- **Payment Card Industry Data Security Standard (PCI DSS),** consist of a set of AWS security best practices controls. Each control applies to a specific AWS resource, and related to one or more PCI DSS requirements. This standard is designed to help you with your ongoing PCI DSS security activities. The controls cannot verify whether your systems are compliant with the PCI DSS standard. They can neither replace internal efforts nor guarantee that you will pass a PCI DSS assessment. Security Hub does not check procedural controls that require manual evidence collection.

AWS Security Hub provides a centralized view for monitoring and managing findings to support multi-account and multi-region configurations, across the below services:

- Amazon GuardDuty
- Amazon Inspector
- IAM Access Analyzer
- Amazon Macie
- AWS Firewall Manager
- Other 3rd Party Partner Solutions

At VMO2, deploying Security Hub Enabler solution will enable Security Hub in all Control Tower managed accounts, with the Security Services account acting as the default Security Hub Administrator.

"AWS Foundational Security Best Practices" and "CIS AWS Foundations" standards will be enabled in all AWS Accounts. "Payment Card Industry Data Security Standard (PCI DSS)" standard can be enabled for accounts that have resources that store, process, and/or transmit cardholder data.

These AWS Security findings will be ingested into VMO2's SIEM solution (Chronicle). Based on the Chronicle documentation is able to integrate using as source S3 bucket, and collect CloudTrail and Cloudwatch Logs. As illustration purpose, the figure below explains how it's possible to use a SharedServices central logging account to later get the data into Chronicle.

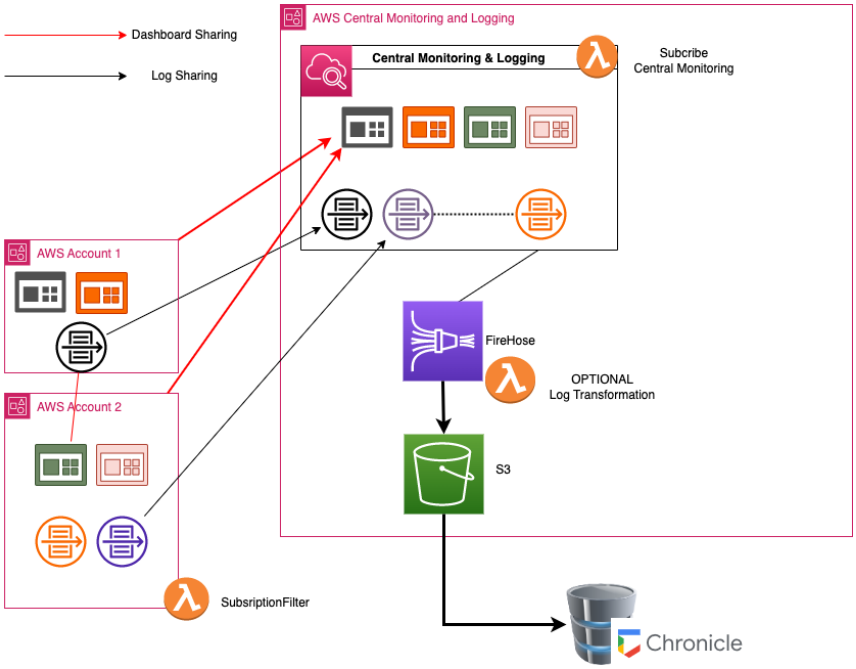


FIGURE 23- CENTRALIZED LOGGING INTEGRATION WITH CHRONICLE

Note – Although the logs are exported to a third party, we recommend to use the CloudTrail Insights capability in CloudTrail to find behavior out of the baseline. This is an automatic process handle by the platform for you. It helps AWS users identify and respond to unusual activity associated with write API calls by continuously analyzing CloudTrail management events.

4.13.2 GuardDuty

Amazon GuardDuty is an intelligent threat detection service that provides customers with an accurate and easy way to continuously monitor and protect their AWS accounts and workloads. GuardDuty analyses billions of events across AWS accounts from AWS CloudTrail (AWS user and API activity in VMO2 accounts), Amazon VPC Flow Logs (network traffic data), and DNS Logs (name query patterns).

Amazon GuardDuty threat detection identifies activity that can be associated with account compromise, instance compromise, and malicious reconnaissance. For example, GuardDuty detects unusual API calls, suspicious outbound communications to known malicious IP addresses, or possible data theft using DNS queries as the transport mechanism. GuardDuty delivers more accurate findings using machine learning enriched by threat intelligence, such as lists of malicious IPs and domains.

At VMO2, the deployed GuardDuty solution will delegate the Security Services/Audit account as the GuardDuty administrator account in all Regions of the AWS Control Tower managed organization. It will add all existing AWS Control Tower accounts as GuardDuty members of the GuardDuty administrator in the same Region and will export GuardDuty findings from the GuardDuty management account in all Regions to a single S3 bucket in the AWS Control Tower log archive account.

4.13.3AWS Config Conformance Packs

A conformance pack is a collection of AWS Config rules and remediation actions that can be easily deployed as a single entity in an account and a Region or across an organization in AWS Organizations, to support multi-accounts & multi-regions.

Conformance packs are created by authoring a YAML template (similar to CloudFormation) that contains a list of AWS Config managed or custom rules and remediation actions.

Templates exist as samples for using AWS Config rules and remediation actions within a conformance pack. They do not guarantee thorough compliance check as is. You need to modify the template by adding more AWS Config rules to meet VMO2 requirements to ensure thorough compliance of your AWS resources.

- Operational Best Practices for AWS Well Architected Security Pillar
- Operational Best Practices for AWS Well Architected Reliability Pillar
- Operational Best Practices for Amazon DynamoDB
- Operational Best Practices for Amazon S3
- Operational Best Practices for AWS Identity and Access Management
- Operational Best Practices for CIS
- Operational Best Practices for NIST CSF
- Operational Best Practices for PCI-DSS
- AWS Control Tower Detective Guardrails

4.14 Tagging

Decision	VMO2 will consider the tagging a priority once the MVP is in place. However the design document will align with the actual Tagging implementation available.

Amazon Web Services allows customers to assign metadata to their AWS resources in the form of tags. Each tag is a simple label consisting of a customer-defined key and an optional value that can make it easier to manage, search for, and filter resources by purpose, owner, environment, or other criteria. AWS tags can be used for many purposes.

Once cloud resources start getting created, it rapidly becomes apparent that keeping track of resources, their purpose, and how they should be tracked for billing is necessary. For this, a tagging strategy should be defined and enforced.

Using tagging within AWS has numerous use cases and benefits and is a generally recommended best practice for all AWS customers.

4.14.1 Tag naming limits and requirements

- The following basic naming and usage requirements apply to tags:
- **Each resource can have a maximum of 50 user created tags.**
- System created tags that begin with aws: are reserved for AWS use, and do not count against this limit. A tag that begins with the prefix “aws:” cannot be edited or deleted.
- For each resource, each tag key must be unique, and each tag key can have only one value.
- The tag key must be a minimum of 1 and a maximum of 128 Unicode characters in UTF-8.
- The tag value must be a minimum of 0 and a maximum of 256 Unicode characters in UTF-8.
- Allowed characters can vary by AWS service. For information about what characters you can use to tag resources in a particular AWS service, see its documentation. In general, allowed characters in tags are letters, numbers, spaces representable in UTF-8, and the following characters: _ . : / = + - @ .
- Tag keys and values are case sensitive. As a best practice, decide on a strategy for capitalizing tags, and consistently implement that strategy across all resource types. For example, decide whether to use Costcenter, costcenter, or CostCenter, and use the same convention for all tags. Avoid using similar tags with inconsistent case treatment.

Note: Some services don't permit tags with an empty value (length of 0).

4.14.2Tagging strategy at VMO2:

Refer to https://docs.aws.amazon.com/general/latest/gr/aws_tagging.html
Refer to this link: https://galactic.virginmedia.co.uk/confluence/display/PCP/AWS+Tagging

Tag	Required	Usage	Value/Type
Name	Mandatory	Technical Tag	String
Application ID	Mandatory	Technical Tag	String
Application Role	Mandatory	Technical Tag	String
Tech Owner	Mandatory	Business Tag	String
Business Owner	Mandatory	Business Tag	String
Environment	Mandatory	Technical Tag	String (e.g Training)
Version	Mandatory	Technical Tag	String: major.minor (e.g. 1.3)
Cluster	Mandatory	Technical Tag	String: resource farm (e.g. IXS)
Project	Mandatory	Business Tag	String: Project ID
CostCenter/BU	Mandatory	Business Tag	String: cost center ID
Security	Mandatory	Automation Tag	List: Encryption, VPC Flow enable,SGs,etc.
Criticality	Mandatory	Security Tag	List: Critical, Major, Moderate, Minor
Confidentiality	Mandatory	Security Tag	List: High, Medium, Low
Compliance	Mandatory	Security Tag	List:PCi-DSS, HIPA, etc..
DataTime	Mandatory	Automation Tags	Date or Time to start,stop, delete or rotate resource.
OptInOptOut	Mandatory	Automation Tags	Boolean – aligned with Escalabilty Groups/Scheduling
Environment	Mandatory	Technical Tag	List: Dev, Test, Prod
Comment	Optional		String
JIRA	Optional		String
Name	Optional		String
Time to Live	Optional		String
Recovery Time Objective	Optional		0
Recovery Point Objective	Optional		0
Proof of Concept	Optional		Boolean (True / False)

4.14.3Tagging Policy

AWS Organizations has the capability to apply a tagging policy. This policy does not ensure that tags are applied but when tags are applied it can ensure they follow a consistent name (casing, spelling, and formatting) as well as that those tags with a defined list of acceptable values can contain only those values.

A JSON document that defines the initial tagging policy at VMO2 will be created. That policy will be deployed to the AWS Organization and activated for the entire organization.

For example, a tag policy can specify that when the CostCenter tag is attached to a resource, it must use the case treatment and tag values that the tag policy defines. A tag policy can also specify that noncompliant tagging operations on specified resource types are *enforced*. In other words, noncompliant tagging requests on specified resource types are prevented from completing. Untagged resources or tags that aren't defined in the tag policy aren't evaluated for compliance with the tag policy. Using tag policies involves working with multiple AWS services:

- Use **AWS Organizations** to manage *tag policies*. When you sign in to the organization's management account, you use Organizations to enable the tag policies feature. You must sign in as an IAM user, assume an IAM role in the organization's management account. Then you can create tag policies and attach them to the organization entities to put those tagging rules in effect.
- Use **AWS Resource Groups** to manage *compliance* with tag policies. When you sign in to an account in your organization, you use Resource Groups to find noncompliant tags on resources in the account. You can correct noncompliant tags in the AWS service where you created the resource.

Example of Tag Enforcement Using SCP – on it, a SCP is applied to an OU in the AWS Organization. The test account is inside of the OU. The SCP that is applied prevents EC2 Instances from being launched if they lack the “Project” and “CostCenter” Tags.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "DenyRunInstanceWithNoProjectTag",
   "Effect": "Deny",
   "Action": "ec2:RunInstances",
   "Resource": [
    "arn:aws:ec2:*:*:instance/*",
    "arn:aws:ec2:*:*:volume/*"
   ],
   "Condition": {
    "Null": {
     "aws:RequestTag/Project": "true"
    }
   }
  },
  {
   "Sid": "DenyRunInstanceWithNoCostCenterTag",
   "Effect": "Deny",
   "Action": "ec2:RunInstances",
   "Resource": [
    "arn:aws:ec2:*:*:instance/*",
    "arn:aws:ec2:*:*:volume/*"
   ],
   "Condition": {
    "Null": {
     "aws:RequestTag/CostCenter": "true"
    }
   }
  }
 ]
}
```

4.14.4Tag Enforcement and Value Enforcement Using AWS Config Rules

The AWS Config Rule REQUIRED_TAGS checks if your resources have the tags that you specify. For example, you can check whether your Amazon EC2 instances have the CostCenter tag. Separate multiple values with commas. You can check up to 6 tags at a time.

Once the Config Rule is activated, it will search for resources with the Tags that were specified when deploying the Rule.

4.14.5 Global Resource Mandatory Tags (Iteration 1)

Label Key	Label Value (Example)	Description
Name	*Sample-hostname-01*	Name based on *Naming Conventions (see AWS Naming)*
Description	Virgin Media SAP DEV Web Server	Short Description of the resource
OU	CDO	Business Unit
BillingCode	12345	Billing Code or Cost Entity that has fiscal responsibility for the resource. The VMo2 cost center MUST be 5 numerical digits.
Department	Data	Where relevant, the workload owner at a department level inside each OU. **Key pairs (added in code at time of onboarding)?**
PrimaryContact	Team	This should be a "group" or "team" rather than a individual
CreateDate	18-11-2022	**This is mandatory for Sandbox , POC deployments only**. This tag to provide a date for when the POC , Sandbox capability is to be closed / ended. (*Funding to be pre-allocated for the lifetime of this ReviewDate - FInOps control)*
ReviewDate	18-11-2022	**This is mandatory for Sandbox , POC deployments only**. This tag to provide a date for when the POC , Sandbox capability is to be closed / ended. (*Funding to be pre-allocated for the lifetime of this ReviewDate - FInOps control)*
Classification	Confidential	Data classification inline with company IT Information Security standards

5 Logging, Monitoring & Operations

5.1 Initial Considerations

The objective of this section is to define the principles that will guide us on the Landing Zone Logging & Monitoring design:

1. Define the Log sources that will generate log data and the respective log retention periods.
2. Define the Logging architecture for log storage and aggregation.
3. Define the Configuration management solution.
4. Define the Security monitoring services that will be used in the Landing Zone.
5. Define the Observability architecture and services.

In order to meet these goals, the following technical decisions are suggested:

- Platform logs from individual accounts will be stored locally and potentially replicated to the central Monitoring and Logging account

- Logs needed locally by application teams will be stored in CloudWatch Logs in the individual accounts. **CloudWatch Logs** will also be used for storing and analyzing workload logs that do not need to be centralized.
    - The default retention period for operational logs can be adjusted on the Log Retention property for each log group, so VMO2 can adjust those based on the service type and criticality. Be sure you got a default, and you enforce it (*cron* Lambda), or you set it explicitly in resource creation time. You can archive them to log duration storage.
- **Amazon GuardDuty** will be used to monitor threats in all Landing Zone accounts. Findings will be aggregated in the GuardDuty delegated administrator instance in the Audit or Security Tooling account.
- **AWS Config** will be used to continuously assess, audit, and evaluate the configurations of the AWS resources. A Config aggregator in the Audit/Security Tooling account will provide an aggregated configuration view of the whole Landing Zone.
- **AWS Security Hub** will be used by application teams in local accounts to manage the security posture of their cloud resources. Security Hub will also be aggregated in the Security Tooling account to give the platform team visibility into the Landing Zone. If it's required aggregated findings can be forwarded to a third-party compliance monitoring solution.
- **AWS CloudTrail** will be used by the security and/or audit teams to enable governance, compliance, and operational and risk auditing of your AWS account. We will consider trails x region, and the CloudTrail logs files will be reviewed via CloudWatch Logs, and stored long term in the central Log account.
- **AWS Inspector** will be used to scan EC2 instances and container images residing on ECR (Container Registry) in searching for vulnerabilities. It will together with Security Hub to publish the findings.

Decision	VMO2 will not consider central logging & monitoring on the MVP. Under consideration for next phases. GuardDuty and AWS Config will be used. Inspector/SecurityHub will be considered minimizing the overlapping with ORCA[3]. As such controls in ORCA CSPM will not be duplicated in Control Tower Security.

5.1.1 Cloud Platform Monitoring and Logging Overview

Monitoring allows you to plan for and respond to potential incidents. The results of those activities are stored in log files; therefore, logging and monitoring are closely related concepts. When you set up your landing zone, one of the shared accounts created is the *log archive* account, dedicated to collecting all logs centrally, including logs for all of your other accounts. These log files allow administrators and auditors to review actions and events that have occurred.

As a best practice, you should collect monitoring data from all of the parts of your AWS solution into your logs, so that you can more easily debug a multi-point failure if one occurs and ensure all the changes are logged. This helps you to ensure that any unexpected change can be investigated and unwanted changes can be rolled back. Based on the path followed to implement the landing zone, you can get the majority of this baseline setup almost automatically using Control Tower vs a more customized approach where you will need to ensure the setup is achieved by other means (.i.e. Terraform, or CloudFormation)

There are two main AWS services that enable you to monitor/log your organization related infrastructure and the activity that happens within it including the automation itself: AWS Cloudtrail and AWS Cloudwatch Events and Logs. On the upcoming sections you will see how these two AWS services are used multiple times from different purposes as they are containers for many individual and grouped information sources.

The following diagram illustrates the high-level design for a baseline/initial logging and monitoring solution. As starting point, each account will leverage its own native CloudWatch functionality to provide visibility of infrastructure, application and security logs and metrics across different environments. This will allow users of each account to

[3] Orca & AWS - https://orca.security/resource/literature/orca-security-aws-brochure/

manage their own logs and metrics while having the capability to integrate with other accounts or systems while enabling the security team to keep the information on 3rd party SIEM or Amazon OpenSearch.

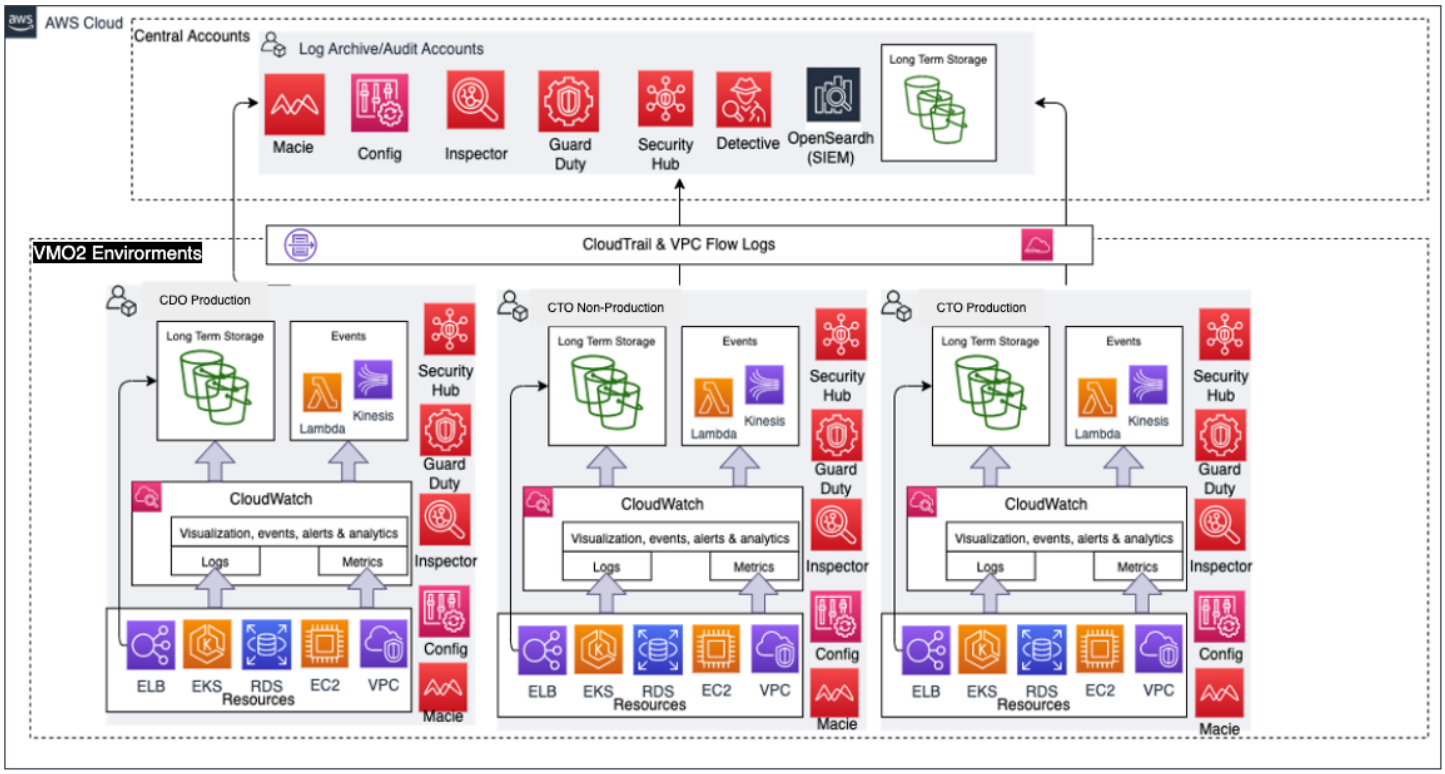


FIGURE 24 - LOGGING AND MONITORING ARCHITECTURE EXAMPLE

5.2 Platform Monitoring

Monitoring is an important part of maintaining the reliability, availability, and performance of your AWS solutions. The following monitoring tools can be used to watch, report when something is wrong, and take automatic actions when appropriate:

- **Amazon CloudWatch** monitors your AWS resources and the applications you run on AWS in real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify.
- **Amazon CloudWatch Events** delivers a near real-time stream of system events that describe changes in AWS resources. CloudWatch Events enables automated event-driven computing, as you can write rules that watch for certain events and trigger automated actions in other AWS services when these events happen.
- **Amazon CloudWatch Logs** enables you to monitor, store, and access your log files. It can monitor information in the log files and notify when certain thresholds are met. We will also archive log data in highly durable storage as we will explain below (Log Archive Account)
- **AWS CloudTrail** captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify (Log Archive Account). You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred (*tracing*).

It's possible to view and query CloudTrail activity on an account through CloudWatch Logs and CloudWatch Logs Insights. This one allows to perform more granular and precise queries than you would normally be able to make using CloudTrail.

Decision	VMO2 will enable CloudTrail and CloudTrail CloudWatch Logs at the Landing Zone creation time.

Security is covered by many services in AWS, but due to the nature if CloudTrail, the usage of the CloudWatch Logs associated to CloudTrail is a complementary source of information to detect anomalies. AWS CloudTrail Insights is a feature of CloudTrail that can be used to identify unusual operational activity in your AWS accounts such as spikes in

resource provisioning, bursts of AWS Identity and Access Management (IAM) activity, or gaps in periodic maintenance activity.

5.3 Platform Logging

We can use the following logs to monitor and audit your landing zone. As such the information can be used as pure logging source or as security event triggering (see next section).

- **AWS Cloud Trail Logs** – it logs the API calls, console access and logins. Logs from all accounts are aggregated on the Log Archive Account. It must be enabled in all the accounts in the landing zone. Logs are stored on an S3 bucket on the Log Archive Account.
- **AWS Config** – it logs all the configuration activity.
- **Amazon CloudWatch Logs** - CloudWatch monitors resources and applications in the environment in real time. IT collects and tracks metrics for the resources and applications. It's recommended to setup CloudWatch for all required AWS resources and stored in S3 buckets provided with the workloads (prefix the bucket).
    - ELB Logs
    - DNS Logs (if Hybrid DNS is used)
    - Application Specific Logs (see internal monitoring & EKS monitoring and Logging).
- **Amazon S3 Access Log** – enable it to provide details for the request made to the S3 bucket. It must be enabled for the buckets used for CloudTrail and AWS Config.
- **VPC Flow Logs**– to capture information about traffic between network interfaces on the VPC. Stored in each member account, for troubleshooting and analysis. The enablement of the VPC Flow Logs must be done using IaC and it must be ready to be used for each VPC in every account. VPC Flows are Locally sent to every CloudWatch in every account. The recommended retention will be three days. Only used when needed. Because the fields on the VPC flows can be customized is a good idea to have predefine different formats with more or less fields to be captured when enabled.
- **Lambda Logs** - reports logging information either generic or instrumented.

In general, any service generating a Log will be able to provide similar set of information.

5.3.1 Log Archive and Audit Accounts

As mentioned on the Account Structure section, there are two accounts that must be taken in consideration during the Landing Zone Build up; the **Log Archive** Account and the **Audit Account**. Refer to the account structure to understand in depth the responsibilities of those accounts. Those accounts play a critical role on the logging related activities involving the Landing Zone governance.

5.3.2 Logging Sources

AWS services provide vast logging capabilities. Correlating collected information among event sources can provide a robust security posture and enhance visibility. The following table contains information about common log sources in AWS:

Logging Source	Log Destination	Format	Delay
CloudTrail	CloudWatch Logs, Amazon S3	JSON	Amazon S3: Within 15 minutes CloudWatch Events: Near real-time
Amazon VPC Flow Logs	CloudWatch Logs, Amazon S3	space-separated string format	< 10 minutes (default) ~1 minute (on demand)

AWS Config	Amazon S3	JSON	Every 6 hours (if there are configuration changes)
Amazon S3 Data Events	CloudTrail	JSON	Near real-time to CloudTrail (see CloudTrail for delay)
AWS Lambda Data Events	CloudTrail	JSON	Near real-time to CloudTrail (see CloudTrail for delay)
AWS Lambda Execution Logs	CloudWatch Logs	CloudWatch Logs log stream	Near real-time
Elastic Load Balancer Logs	Amazon S3	Space-separated string format	Amazon S3: ~ 5 minutes
Amazon RDS Logs	API Logs (in Amazon S3), Database logs (Amazon RDS)	CloudTrail logs (JSON), Database log file (varies by engine)	Near real-time to CloudTrail (see CloudTrail for delay) Database logs: Near real-time

5.4 Incident Response (Recommended Services to consider using)

5.4.1 Security Response Automation

Security response automation is a planned and programmed action taken to achieve a desired state for an application or resource based on a condition or event. AWS provides the following tools to automate aspects of this best practice.

5.4.2 AWS Lambda

Uses the serverless compute service to run code without provisioning or managing servers so you can scale your programmed, automated response to incidents. You can trigger Lambda functions to respond to events logged to CloudWatch and CloudTrail, or published to a Simple Notification Service (SNS) topic.

5.4.3 CloudWatch Events

Delivers a near real-time stream of system events that describe changes in Amazon Web Services (AWS) resources. Using simple rules that you can quickly setup, you can match events and route them to one or more target functions or streams. It becomes aware of operational changes as they occur.

CloudWatch Events responds to these operational changes and takes corrective action as necessary, by sending messages to respond to the environment, activating functions, making changes, and capturing state information.
You can also use CloudWatch Events to schedule automated actions that self-trigger at certain times using *cron* or rate expressions.

5.4.4 AWS Systems Manager, Automation

This Systems Manager capability, lets you simplify common maintenance and deployment tasks of Amazon EC2 instances and other AWS resources. It includes several pre-defined Automation documents that you can use to perform common tasks like restarting one or more Amazon EC2 instances or creating an Amazon Machine Image (AMI). You can create your own Automation documents as well.

- Convert manual and repetitive tasks into automated steps
- Use predefined runbooks or create your own runbooks
- Delegate administration to safely perform operations at scale
- Enable approval steps

5.4.5 AWS Step Functions

A web service that enables you to coordinate the components of distributed applications and microservices using visual workflows. You build applications from individual components that each perform a discrete function, or task,

allowing you to scale and change applications quickly. Provides a reliable way to coordinate components and step through the functions of your application. It automatically triggers and tracks each step, and retries when there are errors, so your application executes in order and as expected, every time. It logs the state of each step, so when things go wrong, you can diagnose and debug problems quickly.

Appendix A - References

Features of key AWS services:

AWS Control Tower: https://aws.amazon.com/controltower/features/
AWS Transit Gateway: https://aws.amazon.com/transit-gateway/features/
AWS Systems Manager: https://aws.amazon.com/systems-manager/features/
Amazon Route53: https://aws.amazon.com/route53/features/
AWS Organizations: https://aws.amazon.com/organizations/features/
AWS Single Sign-On: https://aws.amazon.com/single-sign-on/features/
AWS CloudFormation: https://aws.amazon.com/cloudformation/features/
AWS Service Catalogue: https://aws.amazon.com/servicecatalog/features/
Amazon GuardDuty: https://aws.amazon.com/guardduty/features/
AWS Security Hub: https://aws.amazon.com/security-hub/features/

Additional Reference links:

https://docs.aws.amazon.com/controltower/latest/userguide/sso.html
https://aws.amazon.com/blogs/aws/the-next-evolution-in-aws-single-sign-on/
https://docs.aws.amazon.com/singlesignon/latest/userguide/okta-idp.html
https://www.okta.com/partners/aws/
https://docs.aws.amazon.com/singlesignon/latest/userguide/permissionsetsconcept.html
https://docs.aws.amazon.com/controltower/latest/userguide/guardrails.html
https://docs.aws.amazon.com/directconnect/latest/UserGuide/direct-connect-transit-gateways.html
https://aws.amazon.com/premiumsupport/knowledge-center/dx-configure-dx-and-vpn-failover-tgw/
https://aws.amazon.com/blogs/security/how-to-add-dns-filtering-to-your-nat-instance-with-squid/
https://docs.aws.amazon.com/general/latest/gr/aws-service-information.html
https://aws.amazon.com/blogs/networking-and-content-delivery/integrating-aws-transit-gateway-with-aws-privatelink-and-amazon-route-53-resolver/
https://aws.amazon.com/blogs/security/how-to-delegate-administration-of-your-aws-managed-microsoft-ad-directory-to-your-on-premises-active-directory-users/
https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-automation-multiple-accounts-and-regions.html
https://docs.aws.amazon.com/systems-manager/latest/userguide/what-is-systems-manager.html
https://aws.amazon.com/blogs/mt/centralized-multi-account-and-multi-region-patching-with-aws-systems-manager-automation/
https://docs.aws.amazon.com/systems-manager/latest/userguide/Explorer.html
https://aws.amazon.com/blogs/mt/automating-amazon-guardduty-deployment-in-aws-control-tower/

Appendix B - Terms and Definitions

The table below provides definitions of the terms used throughout this document.

Term	Definition
AD	Active Directory, a directory service and identity store
ADF	AWS Deployment Framework, an extensive and flexible framework to manage and deploy resources across multiple AWS accounts and regions within an AWS Organization
AMI	Amazon Machine Image
API	Application Programming Interface, an interface that allows for programmatic access between applications
AWS	Amazon Web Services
AWS Account	The AWS Account is the identity under which your AWS services and resources run
AWS root account	The primary (root) user associated with an AWS account, will require a unique email address and login credentials
AWS Config	A service to assess, audit, and evaluate configuration of your AWS resources
AWS IAM	AWS Identity and Access Management
AWS Organizations	Account management service that lets you centrally manage multiple AWS accounts
AWS SNS	Amazon Simple Notification Service
AWS SQS	Amazon Simple Queue Service
AWS SSM	AWS Systems Manager, an AWS service that you can use to view and control your infrastructure on AWS
AWS SSO	AWS cloud-based single sign-on service (now Identity Central)
BU	Business Unit
CDK	AWS Cloud Development Kit
CIDR	Classless inter-domain routing, a method for allocating IP addresses and IP routing
CMP	Cloud Management Platform
CNAME	Canonical Name Record
COTS	Commercial Off The Shelf product or service
DNS	Domain Name System
DXGW	AWS Direct Connect Gateway
EC2	Amazon Elastic Compute Cloud, a web service that provides compute capacity in the cloud
ECR	Amazon Elastic Container Registry
ENI	Elastic Network Interface
Federation	The linking of an identity store across multiple identity management systems
Guardrail	A high-level rule that provides ongoing governance for your AWS environment, these can be SCP's or Config rules
IAC	Infrastructure as Code
IAM Role	A set of permissions without credentials that are temporarily assumed by an IAM user to perform a task
IAM User	An identity representing a person or service used to interact with AWS services by using the associated credentials
IdP	Identity Provider
IPSEC	Internet Protocol Security, a secure network protocol suite that authenticates and encrypts the packets of data between two computers
JSON	JavaScript Object Notation, a format for storing and transporting data
MFA	Multi-Factor Authentication, a user must provide two or more pieces of evidence to verify their identity
MPLS	Multiprotocol Label Switching
MX	A mail exchanger record
NACL	Network Access Control List, a layer of security for your VPC that acts as a firewall for controlling traffic in and out of subnets
NAT	Network address translation
NFVO	Network Function Virtualization Orchestrator
Organizational Unit (OU)	A logical way to group AWS accounts together for administrative purposes
PHZ	Private Hosted Zone
Principal	A person or application that uses an IAM role to sign in and make requests to AWS
Resource	In AWS a resource is an entity that you can work with, for example EC2 instances
SaaS	Software as a Service

Term	Definition
SAML	Security Assertion Markup Language (SAML) is a standard that allows identity providers to pass authorization to a service provider
Service Control Policy (SCP)	An organizational policy to manage permissions for all accounts in your organization
SSO	Single Sign On – Service that enables a user to use a single set of credentials for multiple services
Tags, Tagging	A label that can be assigned to an AWS resource for access management, billing, etc.
TGW	Transit Gateway, a network transit hub that you can use to connect your virtual private clouds and on-premises networks
VPC	A Virtual Private Cloud. A virtual network separation in AWS into which resources such as virtual machines can be launched

END of the DOCUMENT