Of course. Here is the comprehensive multi-cloud Identity and Access Management (IAM) strategy, synthesized from the provided documentation and validated against current industry best practices.

## **Multi-Cloud IAM Strategy: Executive Overview**

This document outlines the unified Identity and Access Management (IAM) strategy for securing and managing identities across Amazon Web Services (AWS), Microsoft Azure/Entra ID, and Google Cloud Platform (GCP). Our vision is to establish a unified, automated, and compliance-driven IAM framework that enforces the principle of least privilege by default. This strategy directly supports our organizational goals by mitigating cyber risk, enhancing operational efficiency through automation, and ensuring adherence to our regulatory obligations, including **NIST 800-53**, **FFIEC CAT**, and **CSA STAR**.

The core of this strategy is to move from a fragmented, partially manual state to a cohesive model built on the following principles:

* **Centralized Identity:** Establish a single Identity Provider (IdP) as the source of truth for all user authentication to streamline access and strengthen auditing.
* **Automated Governance:** Leverage Infrastructure as Code (IaC) and Policy as Code (PaC) to automate the lifecycle of IAM resources, enforce security policies, and eliminate manual errors.
* **Dynamic & Granular Access:** Implement Just-in-Time (JIT) access and context-aware controls to ensure users and services have only the permissions they need, for only as long as they need them.
* **Continuous Visibility & Monitoring:** Utilize an Identity Security Posture Management (ISPM) platform to gain deep visibility into all identities (human and non-human) and their effective permissions, ensuring continuous compliance and rapid risk detection.

This strategy will be implemented in realistic phases, acknowledging the capacity of our two-person team, to deliver foundational controls first and build toward a mature, proactive security posture.

## **Universal Cross-Cloud IAM Strategy**

These universal principles and requirements form the foundation of our IAM program and will be applied consistently across AWS, Azure, and GCP to create a cohesive security posture.

### **Identity and Federation**

A centralized identity authority is the cornerstone of a secure multi-cloud strategy, as it reduces the attack surface and simplifies compliance.

* **Single Identity Provider (IdP):** **Okta** will be established as the single, authoritative IdP for all human user authentication into AWS, Azure, and GCP. The in-progress migration from Ping Identity for AWS access will be prioritized to eliminate the complexity and audit overhead of a split-IdP model.
* **Authoritative Identity Source:** **On-premises Active Directory**, governed by **SailPoint IdentityIQ**, will remain the authoritative source for identity lifecycle management (joiner, mover, leaver) and group-based role assignments.
* **Standardized Federation Protocols:** **SAML 2.0** and **OpenID Connect (OIDC)** will be the standard protocols for federating identities from Okta to the cloud platforms, ensuring secure and consistent single sign-on (SSO) and multi-factor authentication (MFA) enforcement.
* **Workload Identity Federation:** We will adopt cloud-native workload identity federation where possible. This practice allows workloads (e.g., CI/CD pipelines, Kubernetes pods) to access cloud resources using short-lived tokens instead of long-lived static credentials, significantly improving security.

### **Authorization and Governance**

Our goal is to enforce least privilege access dynamically and at scale.

* **Hybrid Access Model:** We will implement a hybrid authorization model combining Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC).
  + **RBAC** will provide a baseline of permissions based on a user's job function.
  + **ABAC** will be used to create fine-grained, dynamic permissions based on context like resource tags, time of day, or user location.
* **Just-in-Time (JIT) Access:** Privileged access will be granted exclusively through a JIT model. Instead of standing privileges, users will request temporary, time-bound elevation into roles, subject to approval. The in-progress "JIT via SailPoint" initiative is the primary vehicle for this.
* **Identity Security Posture Management (ISPM):** The onboarding of **Veza** is critical and will serve as our central plane of glass for identity governance. It will connect identities to data, providing visibility into effective permissions for both human and non-human identities and supercharging our SailPoint investment by covering modern cloud and SaaS environments.
* **Access Reviews:** User access reviews will continue to be conducted and automated through SailPoint and augmented by insights from Veza to identify and remove excessive or unused permissions (privilege creep).

### **Secrets Management**

We will move from a fragmented, manual approach to a unified and automated secrets management lifecycle.

* **Centralized Secrets Orchestration:** While CyberArk is the incumbent tool, we will establish a unified strategy for it to manage secrets across all cloud platforms, not just on-premises. This involves leveraging features like **CyberArk's Secrets Hub** to integrate with native cloud vaults (AWS Secrets Manager, Azure Key Vault) or exploring a dedicated secrets orchestration platform like **Pulumi ESC**.
* **Automated Rotation:** All secrets—including access keys, passwords, and API tokens—must be rotated automatically on a defined schedule (e.g., every 90 days). This process will be automated via IaC and integrated into our CI/CD pipelines.
* **Elimination of Static Credentials:** Long-lived static credentials for service accounts will be deprecated in favor of workload identity federation and other short-lived token-based mechanisms.

### **Automation and Code-Driven IAM**

Manual processes will be systematically eliminated in favor of a fully automated, auditable, and code-driven approach.

* **Infrastructure as Code (IaC):** **Terraform** will be the exclusive tool for provisioning all IAM resources, including roles, policies, service accounts, and permissions. This ensures configurations are version-controlled, repeatable, and auditable.
* **Policy as Code (PaC):** We will integrate PaC tools like **Open Policy Agent (OPA)** or **HashiCorp Sentinel** into our CI/CD pipelines. This allows us to enforce preventative guardrails by validating Terraform plans against security and compliance policies *before* deployment, blocking misconfigurations like overly-permissive roles or incorrect tagging.
* **Standardized Tagging:** The existing tagging policy (application\_name, environment, owner) is foundational and will be strictly enforced via PaC. Tags are critical for cost allocation, resource ownership, and enabling automated governance.

### **Auditing and Monitoring**

Centralized logging provides the data; proactive monitoring provides the intelligence.

* **Centralized Logging:** All cloud-native logs (e.g., AWS CloudTrail, Azure Monitor Logs, GCP Cloud Audit Logs) will continue to be streamed to **Splunk Cloud** for centralized analysis and retention.
* **Proactive Monitoring & Alerting:** Using data from Splunk and Veza, we will develop automated alerts for suspicious IAM activity, such as privilege escalations, use of dormant accounts, or configuration drift from our IaC baseline.
* **IAM Metrics & Dashboards:** We will develop executive and operational dashboards to track Key Performance Indicators (KPIs) for our IAM posture, including JIT access usage, standing privilege reduction, and access review campaign progress. While Splunk can be used for this, we will leverage the specialized, identity-centric analytics within Veza to accelerate this goal.

## **Implementation Roadmap and Phasing**

This strategy will be executed in three distinct 6-month phases to ensure feasibility for a two-person team and to deliver incremental value.

### **Phase 1 (Months 0-6): Foundational Controls & Visibility**

*Goal: Establish a unified identity baseline, gain comprehensive visibility, and codify the most mature environment.*

1. **Complete Ping to Okta Migration:** Finalize the move of AWS federation to Okta, establishing it as the single IdP for all cloud platforms.
2. **Onboard and Integrate Veza:** Fully deploy Veza and connect it to AWS, Azure, GCP, and SailPoint. Use it to create an initial, comprehensive map of all identities and their permissions to establish a baseline security posture.
3. **Codify Azure IAM with Terraform:** Prioritize bringing the manual Azure environment under IaC. Develop Terraform modules for provisioning Azure roles and service principals, and transition the manual ServiceNow workflow to a PR-based process.
4. **Launch Pilot JIT Program:** Implement the first JIT access workflow using SailPoint, targeting a set of high-risk privileged roles in AWS.
5. **Develop Initial IAM Dashboards:** Use Veza's out-of-the-box capabilities to create initial dashboards for tracking standing privileged access and non-human identity risks.

### **Phase 2 (Months 7-12): Automation & Expansion**

*Goal: Automate critical security workflows and expand consistent controls across all clouds.*

1. **Implement Secrets Management Automation:** Begin the unification of secrets management. Select a central orchestration strategy (e.g., CyberArk Secrets Hub) and implement automated rotation for a pilot set of service account credentials in AWS.
2. **Implement Policy-as-Code (PaC) Guardrails:** Integrate a PaC tool (e.g., OPA) with the AWS and Azure Terraform pipelines to enforce tagging and prevent the creation of overly-permissive IAM roles.
3. **Expand JIT Access:** Broaden the JIT program to cover all administrative access in AWS and begin a pilot for Azure PIM (Privileged Identity Management) integrated with SailPoint.
4. **Refine Role Design with Role Mining:** Use data from Veza and native cloud tools (like AWS IAM Access Analyzer) to identify and right-size over-permissioned roles in AWS, submitting PRs to reduce permissions.
5. **Automate GCP IAM:** Bring the small GCP environment into the same IaC and governance model, ensuring all three clouds are managed consistently through Terraform.

### **Phase 3 (Months 13-18): Maturity & Optimization**

*Goal: Achieve a proactive, optimized, and telemetry-driven security posture.*

1. **Adopt Workload Identity Federation:** Refactor key applications in AWS and GCP to use workload identity federation instead of static service account keys, eliminating a major class of secrets.
2. **Full Secrets Orchestration:** Complete the rollout of the automated secrets management solution across all three clouds for both human and non-human identities.
3. **Telemetry-Driven Role Optimization:** Mature the role-rightsizing process into a continuous, automated feedback loop where usage data automatically suggests permission reductions.
4. **Mature Passwordless & Continuous Access Evaluation:** Pilot passwordless (FIDO2) authentication through Okta for high-risk users. Investigate and implement Continuous Access Evaluation (CAE) in Azure to enable real-time session revocation based on risk signals.
5. **Refine and Automate Compliance Reporting:** Automate the generation of evidence for NIST, FFIEC, and CSA STAR controls using data from Veza, Splunk, and SailPoint.

## **Confluence Strategy Pages**

### **Parent Page: Multi-Cloud IAM Strategy**

*(This executive summary and the preceding sections would serve as the main page)*

### **Child Page: AWS IAM Strategy**

# **AWS Identity and Access Management (IAM) Strategy**

This document details the specific strategy for managing identities and permissions within our Amazon Web Services (AWS) environment. It aligns with the universal cross-cloud IAM principles while addressing the unique features of the AWS platform.

## **1. Identity & Federation**

* **Identity Provider:** User access to the AWS console is federated from **Okta** using **SAML 2.0**. All MFA enforcement is handled by Okta.
* **Centralized Access Portal:** Users will access their AWS roles through **AWS IAM Identity Center** (formerly AWS SSO), which will be configured to use Okta as its external identity provider. This provides a unified portal for users to assume roles across our multi-account structure.
* **Role Assumption:** Roles are mapped from **Active Directory groups** managed in SailPoint. Users assume roles via STS, receiving temporary credentials.

## **2. Authorization & Governance**

* **IAM Policies:** Permissions are defined using **identity-based** and **resource-based** policies written in JSON. Policies will be tightly scoped and follow the principle of least privilege.
* **Privileged Access:** All privileged access (e.g., administrative rights) is granted via **Just-in-Time (JIT) elevation** through SailPoint. There will be zero standing privileged access.
* **Service Control Policies (SCPs):** Within AWS Organizations, SCPs will be used to set broad preventative guardrails for all accounts, such as restricting which regions can be used or denying the ability to disable security services like CloudTrail.

## **3. Automation & Code-Driven IAM**

* **Infrastructure as Code (IaC):** All IAM resources—including roles, policies, instance profiles, and IAM Identity Center permission sets—are managed exclusively via **Terraform**. Changes are made via pull request to a central repository and require peer review.
* **Policy as Code (PaC):** An OPA/Sentinel pipeline integration will automatically check all Terraform PRs to enforce:
  + Required tags (owner, application\_name, environment) are present.
  + IAM policies do not contain wildcards (\*) on actions or resources.
  + Cross-account trust policies are restricted to approved accounts.

## **4. Non-Human & Workload Identity**

* **Service Accounts:** Service account access keys are deprecated.
* **Workload Identity Federation:** For workloads running outside AWS (e.g., in GCP or GitHub Actions), we will use **IAM Roles Anywhere** or **OIDC Federation** to allow them to assume roles without long-lived keys.
* **EC2/ECS/Lambda:** Workloads running on AWS compute services will use **IAM Roles for Service** (e.g., instance profiles) to obtain permissions dynamically.

## **5. Secrets & Monitoring**

* **Secrets Management:** **AWS Secrets Manager** will be used for storing and rotating credentials native to AWS services. These secrets will be managed and orchestrated centrally via our enterprise secrets solution.
* **Monitoring:** **AWS IAM Access Analyzer** will be used continuously to identify unused permissions and generate findings for our role-rightsizing process. All **CloudTrail** logs, especially IAM activity, are forwarded to Splunk for centralized monitoring.

### **Child Page: Azure (Entra ID) IAM Strategy**

# **Azure (Entra ID) IAM Strategy**

This document outlines the strategy for managing identities and permissions within Microsoft Azure and Entra ID. It translates our universal IAM principles into the Azure-specific control plane.

## **1. Identity & Federation**

* **Identity Provider:** User sign-in to the Azure portal and services is federated from **Okta** using **SAML and OIDC**. MFA is enforced by Okta.
* **Identity Source:** Our on-premises **Active Directory** is synchronized with **Microsoft Entra ID** via Entra Connect. Entra ID serves as the foundational identity store for Azure.
* **Guest Access:** External user collaboration (B2B) will be managed through **Entra ID B2B**, with defined guest user policies and access reviews.

## **2. Authorization & Governance**

* **Authorization Model:** We use Azure's native **Role-Based Access Control (RBAC)** model. Roles are assigned at specific scopes: Management Group, Subscription, Resource Group, or individual Resource.
* **Privileged Identity Management (PIM):** **PIM** is the mandated mechanism for all privileged access in Azure. Eligible users will request JIT activation into roles like "Global Administrator" or "Subscription Owner." These requests will require justification and approval, creating a full audit trail.
* **Continuous Access Evaluation (CAE):** We will enable CAE to allow real-time enforcement of policy changes. For example, if a user's account is disabled, CAE can terminate their active Azure session almost instantly, rather than waiting for a token to expire.

## **3. Automation & Code-Driven IAM**

* **Infrastructure as Code (IaC):** The current manual ServiceNow process is being deprecated. All Azure RBAC role assignments, PIM configurations, and service principal/managed identity creations will be managed exclusively via **Terraform**.
* **Policy as Code (PaC):** Terraform deployments for Azure will be validated by an OPA/Sentinel pipeline to ensure:
  + Role assignments are not made at the root scope.
  + Custom roles adhere to least privilege principles.
  + Required resource tags for ownership are present.

## **4. Non-Human & Workload Identity**

* **Managed Identities:** For Azure-native services (like VMs, App Service, Functions), we will use **Managed Identities** (both system-assigned and user-assigned) to authenticate to other Azure resources without needing to manage credentials in code.
* **Service Principals:** Service principals will be used for applications and CI/CD pipelines. Their credentials (secrets or certificates) will be stored and rotated by our central secrets management solution.
* **Workload Identity Federation:** For external workloads (e.g., GitHub Actions), we will use **Workload Identity Federation** in Entra ID to enable secure, credential-less deployments.

## **5. Secrets & Monitoring**

* **Secrets Management:** **Azure Key Vault** will be the primary repository for application secrets and certificates within Azure. It will be integrated with our central secrets orchestration platform for management and rotation.
* **Monitoring:** All **Entra ID sign-in logs, audit logs, and provisioning logs** are exported to Splunk for security analysis. We will leverage Azure Monitor and Microsoft Sentinel for real-time alerting on anomalous IAM activity.

### **Child Page: Google Cloud (GCP) IAM Strategy**

# **Google Cloud (GCP) IAM Strategy**

This document details the strategy for managing identities and permissions within our Google Cloud Platform (GCP) environment. While our GCP footprint is currently small, it will be governed by the same rigorous standards as our other cloud platforms.

## **1. Identity & Federation**

* **Identity Provider:** User authentication is federated from **Okta** via **SAML 2.0**.
* **Identity Federation:** We use **Workforce Identity Federation** to allow users from Okta to access GCP resources. This maps external identities to GCP service accounts, avoiding the need to sync user identities into GCP's Cloud Identity.

## **2. Authorization & Governance**

* **Authorization Model:** GCP's IAM model is based on binding principals (users, groups, service accounts) to roles on specific resources (Organization, Folder, Project, or service-level). We will use a combination of **predefined roles** for common tasks and highly-scoped **custom roles** for least-privilege access.
* **Resource Hierarchy:** Permissions will be inherited down the GCP resource hierarchy. We will grant roles at the lowest possible level (e.g., Project or Resource Group) rather than at the Organization level to contain blast radius.
* **Privileged Access:** JIT access will be implemented via integration with our central IAM governance tools, allowing temporary elevation to privileged roles like "Project Owner."

## **3. Automation & Code-Driven IAM**

* **Infrastructure as Code (IaC):** All GCP IAM bindings, custom roles, and service accounts are managed exclusively via **Terraform**.
* **Policy as Code (PaC):** The same OPA/Sentinel pipeline will validate our GCP Terraform code to ensure:
  + The "Project Owner" role is used sparingly.
  + Service account keys are not created (see below).
  + Required labels (GCP's equivalent of tags) are applied.

## **4. Non-Human & Workload Identity**

* **Service Account Keys:** The creation of static JSON service account keys is **strictly forbidden** by an Organization Policy. They are a significant security risk and have been replaced by modern alternatives.
* **Workload Identity Federation:** This is the primary method for workloads running outside GCP (e.g., in AWS, Azure, or CI/CD pipelines) to authenticate to GCP. It exchanges a trusted external token for a short-lived GCP access token.
* **Attached Service Accounts:** For workloads running on GCP compute (e.g., GCE, GKE), we will attach service accounts directly to the resources, allowing them to inherit identity and authenticate automatically using the metadata server.

## **5. Secrets & Monitoring**

* **Secrets Management:** **Google Secret Manager** will be used to store secrets required for applications running in GCP. It is integrated with our central secrets orchestration platform for rotation and management.
* **Monitoring:** **Cloud Audit Logs**, particularly admin activity and data access logs, are exported to Splunk. We use **IAM Recommender** to identify excessive permissions on roles and service accounts, providing insights for our role-rightsizing initiatives.