Strategy to manage machine identities in DevOps & Cloud environments

This paper introduces a 3-tiered strategy for enterprises to manage machine identities as part of their digital transformation initiatives. The foundational principle is that security must be a shared concern between InfoSec, Platform, and Development/Deployment teams. This unifying implementation strategy is needed to address requirements of all stakeholders. InfoSec teams should implement an identity service pro-actively enforcing security policies in an automated manner. Platform teams should utilize platform native plugins or tools to integrate with the identity service provided by InfoSec teams and establish a downstream identity service that manages identities within the boundary of their managed platforms. In the end, Development/Deployment teams should use their existing workflows to request identities ensuring consistent security across the teams.

Table of Contents

Document Control

Version Control

Version	Date	Author	Description
v0.1	2021 03 27	Faisal Razzak	Version 0.1 released

Background

Around the globe, enterprises are digitally transforming their organizations to meet their business objectives to drive innovation to meet future business needs. COVID-19 has only provided impetus to accelerate these efforts. The backbone of this digital transformation depends upon modernizing IT infrastructure and enabling developers to quickly develop solutions for problems. Thanks to the availability of hybrid-cloud environments, cloud-agnostic environments and wide adoption of DevOps operating models, accelerating application deployment is easier than ever before.

Though digital transformation has enabled enterprises to innovate faster than ever before and has unleashed people's ingenuity to solve complex business ideas, it has also created security governance problems for resource-constrained InfoSec teams. Adding to the complexity of these initiatives, there are usually multiple teams involved, each having different focus and objectives: InfoSec, Development/Deployment, and Platform teams.

Security teams need to explore and implement new strategies to keep up with digital transformation efforts that development and operations teams have already adopted. This means they need to explore new security strategies and built-in security tools/plugins that will allow them to easily integrate security measures into DevOps processes.

Platform teams are responsible for managing the enterprise's footprint on public cloud providers and cloud-agnostic platforms. Ensuring security governance is a key requirement for these all teams.

Platform Teams

Broadly speaking, Platform teams fall into one of two categories.

First, the Cloud Operations team is responsible for architecting, deploying, and managing enterprise's footprint on public cloud providers like AWS, Azure or GCP. Second, the SRE or Infrastructure team is responsible for architecting, deploying, and managing enterprise's footprint on traditional data centers and cloud-agnostic platforms like OpenShift or Kubernetes or Serverless.

Regardless of how an enterprise categorizes their platform team, these teams have a responsibility to provide a secure, highly available and easy-to-use environment for different business units within the enterprise. To help achieve this, Platform teams are providing services for the Development & Deployment teams which will help these teams to focus more on solving problems in agile and secure manner.

These Platform teams are also providing a spectrum of services that includes handling ingress/egress traffic, providing network isolation for applications, hosting and storage services, deploying observability through service meshes, identity and access management to the platform. In addition to these services, keeping in sync with the security procedures defined by the InfoSec team is a key requirement and a challenge for Platform teams.

Development & Deployment Teams

For businesses, the ability to respond to current and future business objectives depend upon how rapidly their Development & Deployment teams can develop solutions to meet objectives. On one hand, Developer workflow enables to develop, build, test and release applications using CI/CD (Continuous Integration and Continuous Delivery) pipelines. On the other hand, Deployment workflow enables to consistently deploy applications from sandbox to production environments using CD (Continuous Deployment) pipelines. InfoSec teams are looking to organically embed security in every aspect of developer and deployment workflows. Hence, we hear terms like DevSecOps or Shifting security to the left. DevSecOps deal with several aspects like security testing, scanning, and hardening deployments, ensuring trust and integrity by assigning identities.

Security - A Shared Concern

Security is a shared concern between InfoSec, Platform, and Development/Deployment teams. Every enterprise defines a security policy outlining a set of security controls designed to meet regulatory compliance requirements, standards, baselines and guidelines to be followed. InfoSec is responsible for implementing and enforcing these controls. Traditionally, the enforcement has been achieved by introducing manual approvals through a change control process. Though the traditional enforcement process worked in the past, two factors are contributing to changing old notions and pushing InfoSec team to rethink enforcement process. Those factors are:

- For InfoSec community, Identity is becoming the new security parameter and management of identities form the core component of any approach implementing modern trust architecture. NIST SP 800-207 also advocates an identity centric architecture.
- Traditional development & deployment lifecycle was based on the principles of systems engineering but modern lifecycle is based on the principles of agility and velocity to meet ever evolving and growing business requirements.

Following sections further outline specific security management requirements by different teams for managing machine identities in modern IT infrastructures.

InfoSec Teams

- Governance
- Policy enforcement
- Visibility
- Reporting
- Notifications
- Certificate Ownership
- Accountability
- Change Control
- Crypto agility
- Scalability

Platform Teams

- Self-Service Platform
- Built-in integrations

Development & Deployment Teams

- Automated Certificate lifecycle

Proposed Strategy

I suggests that enterprises address security using a 3-tiered strategy. We explain the above-mentioned strategy by focusing on X.509 certificate (hereafter: certificate) as a machine identity. However, the strategy can be adopted for other machine identities like SSH keys and API/Access Tokens.

Certificates are used to identify different entities such as person, organization, account, process, workloads, or device by using the associated private key. The use of certificates spread from traditional to modern IT infrastructures. Traditionally, network devices have used certificates to enable secure communication using TLS protocol, Operating system and browsers have used certificates to provide root-of-trust, code signing systems have used certificates to digitally sign and verify artifacts to establish the authenticity of the author and integrity of the artifacts. Modern runtime frameworks like SPIFFE/SPIRE and Service Meshes have also recognized the need to use certificates to establish identity for workloads running in cloud-agnostic environments like Kubernetes and Serverless. This entails that enterprises will see a significant growth of certificate consumption in coming years. Managing certificates had been a challenge for InfoSec teams in the past and this challenge will only get bigger as enterprises transition to more modern IT infrastructures.

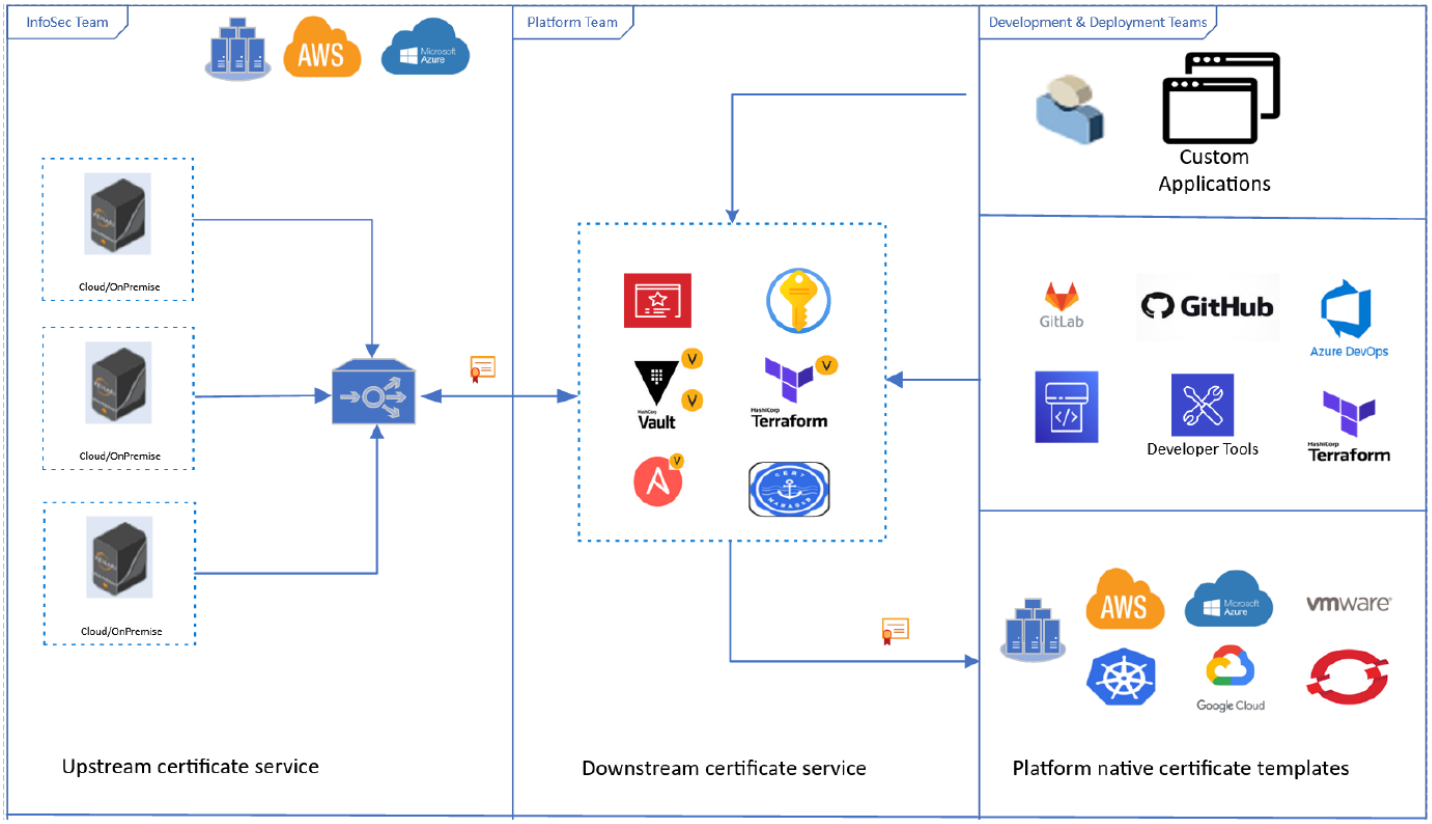

Figure 1: Strategy

This is a depiction of the proposed strategy. InfoSec team establishes upstream certificate service using Venafi's Trust Protection Platform (TPP). Platform teams establish downstream certificate services using native certificate management services like AWS ACM and Azure Key Vault, cloud-agnostic platforms like Kubernetes and HashiCorp Vault, and infrastructure and configuration management tools like Terraform and Ansible. Development & Deployment teams use certificates in custom applications, CI/CD pipelines, infrastructure and cloud platforms.

InfoSec team should act as a service provider and establish an upstream certificate service for different teams providing following features:

- Provide support to enforce security policies in a pro-active and automated manner.
- Provide altering and notification capabilities.
- Provide support to establish ownership of certificates.
- Provide logging and traceability support to establish accountability.
- Provide crypto agility.

- Provide different integration methods ranging from APIs, SDKs to Platform native plugins.
- Provide support for approval process to meat change control board requirements.
- Provide high-availability and the ability to scale based on demand from consumers.

Like any other InfoSec program in an enterprise, broadcasting and communicating information to different business units is a key aspect of establishing a service delivery model. InfoSec team should share information about the availability of the upstream certificate service to Platform teams. In addition, developing platform specific integration guidelines are also important to help bootstrap Platform teams.

While a Platform team should act as a consumer of the upstream certificate service, established by InfoSec team, it should also establish a downstream certificate service using different platform native integration methods. On one hand, the downstream certificate service will enable InfoSec team to pro-actively enforce security policies within the platform, it will also allow Development & Deployment teams to easily retrieve certificates compliant with company's security policies and take the burden off from the Platform team to manage the lifecycle of certificates

The downstream certificate service can operate in either pull integration model or push integration model.

- Pull integration model: Pull integration model is where the upstream certificate service is only acting as a source of policy compliant certificates. Any certificate request originating through the platform is routed to upstream certificate service and the certificate is retrieved. This orchestration process is primarily handled by the downstream certificate service. Pull integration model is suitable for teams that are managing modern cloud-agnostic platforms like OpenShift or HashiCorp Vault, or where deployment of environments and applications is handled by modern Infrastructure and configuration managements tools like Terraform and Ansible.
- Push integration model: Push integration model is where the upstream certificate service can push certificates to the downstream certificate service. This orchestration process is handled by the upstream certificate service. Push model is suitable for teams managing public cloud infrastructure like AWS, Azure, GCP and where cloud-provider native services are used heavily by Development & Deployment teams.

For Platform teams, broadcasting and communicating information to Development & Deployment teams is critical to enable self-service. The availability of a downstream certificate service and development of platform specific certificate templates can help Development & Deployment teams to organically embed security in existing developer and deployment workflows.

While a Development & Deployment team should act as a consumer of the downstream certificate service, it should disband out-of-band certificate request practices and only use platform specific certificate templates. This ensures that all certificates that have been issued are compliant with enterprise's security policies and the burden of managing certificate lifecycle is offloaded to downstream certificate service.

Implementations

1. Security Strategy for Microsoft Azure KeyVault
2. Security Strategy for AWS ACM
3. Security Strategy for HashiCorp Vault
4. Security Strategy for Kubernetes
5. Security Strategy for IaC

Security Strategy for Microsoft Azure KeyVault

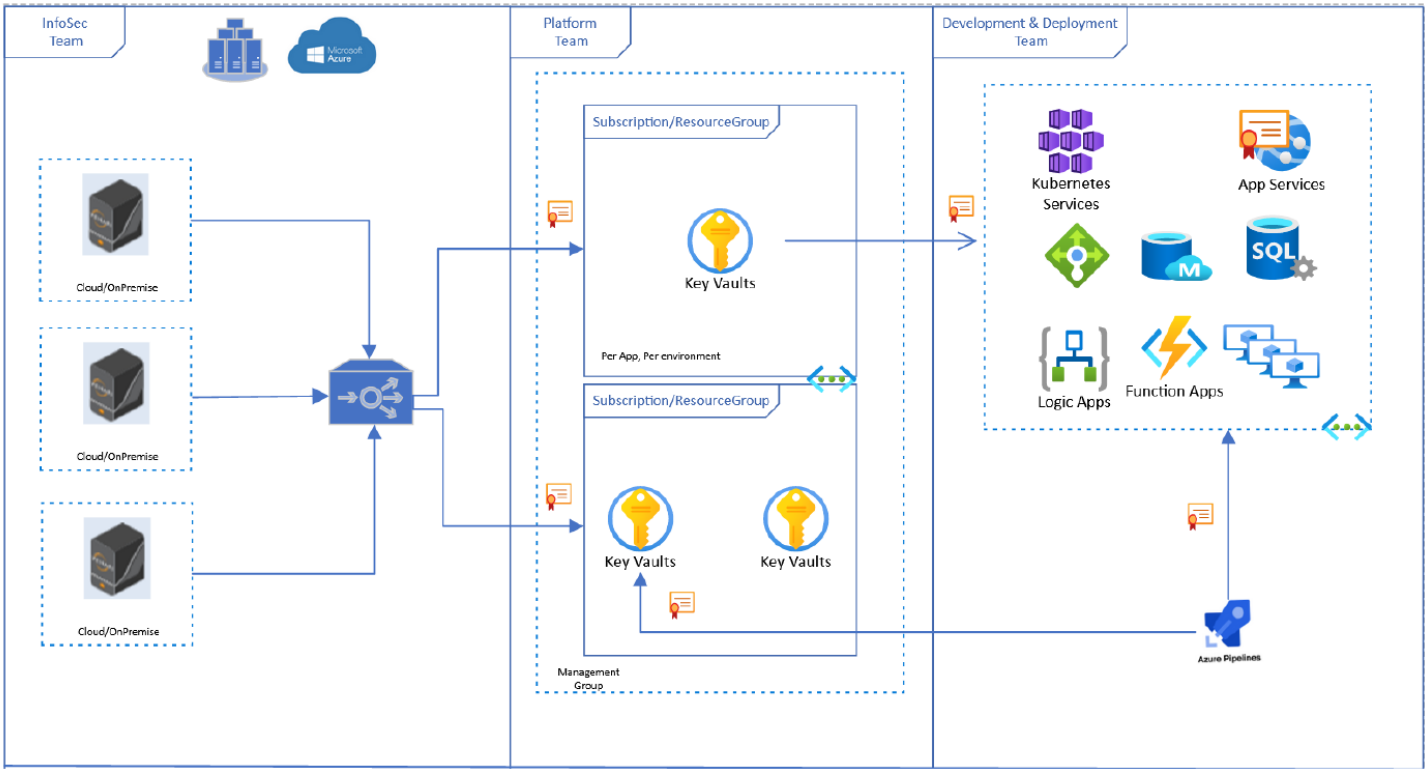

Figure 2: Strategy for an Azure Key Vault using a push integration model

The above diagram shows a depiction of the proposed strategy for AKV using push integration model for securely getting policy compliant certificates from upstream certificate service. InfoSec team provides upstream certificate service using Venafi's TPP that orchestrates the lifecycle of the certificate from request, renewal, revocation to provisioning certificates. Once the certificate is provisioned to AKV, different Azure-native services can consume certificate in AKV using URIs. For applications deployed in Azure but not using Azure-native services, Azure DevOps pipelines can be used to access and consume certificates from AKV. For a Platform team managing Azure cloud deployment introducing AKV as a downstream service is recommended where the use of Azure-native services is understood to be rising across different Development & Deployment teams. Acting as a downstream service, AKV will only handle the deployment of certificates to Azure-native services. The provisioning and management of certificates along with policy enforcement is done through upstream certificate service established by the InfoSec team. It will allow the Platform team to establish AKV as a single source of certificates in Azure and cater to different Development & Deployment teams using Azure-native services.

Security Strategy for AWS ACM

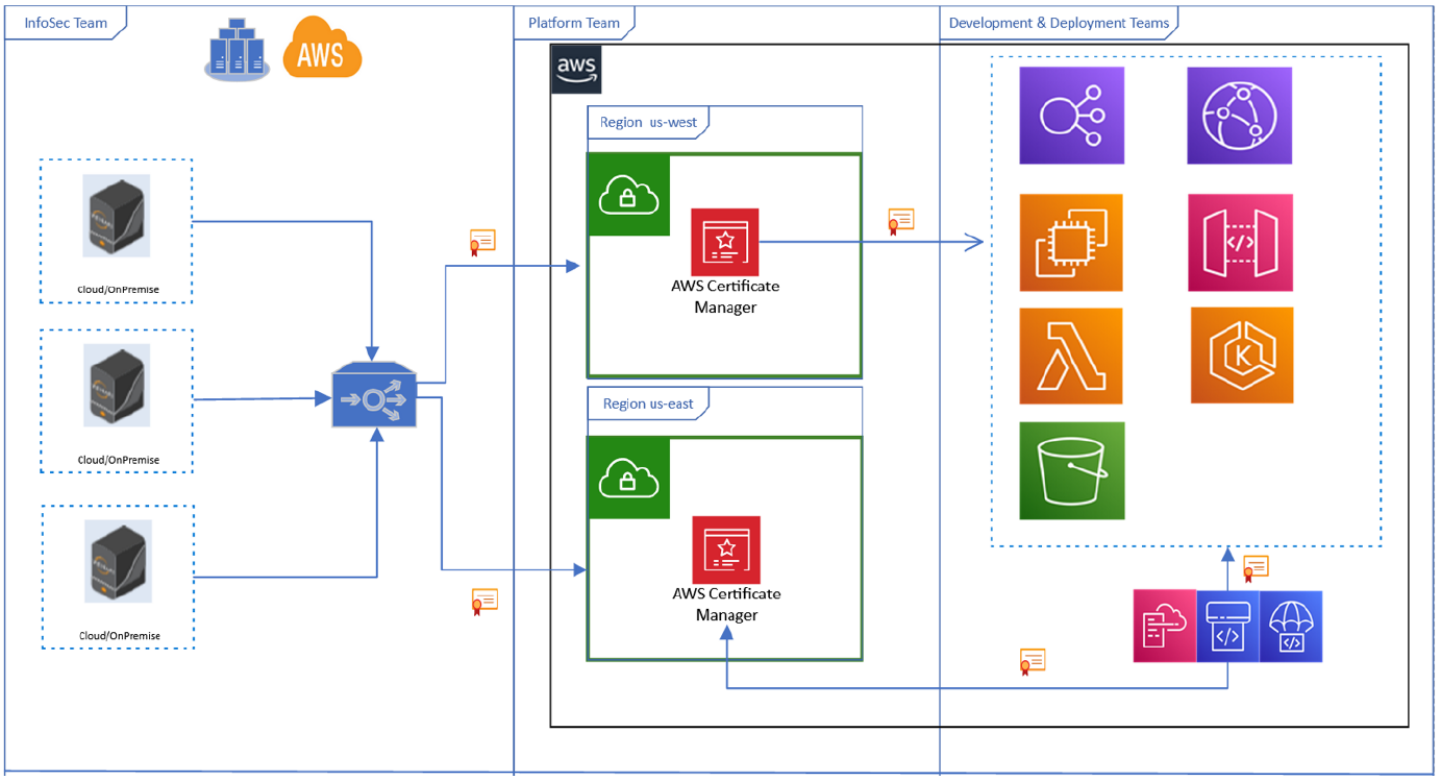

Figure 3: Security Strategy for Amazon ACM using a push integration model

The above diagram shows a depiction of the proposed strategy for ACM using push integration model for securely getting policy compliant certificates from upstream certificate service. InfoSec team provides upstream certificate service using Venafi's TPP that orchestrates the lifecycle of the certificate from request, renewal, revocation to provisioning certificates. Once the certificate is provisioned to ACM, different AWS-native services can consume certificates from ACM. Code Deploy, Code pipelines or CloudFormation services can be used for applications deployed in AWS but not using AWS native services. For a Platform team managing AWS cloud deployment introducing ACM as a downstream service is recommended where the use of AWS-native services is understood to be rising across different Development & Deployment teams. Acting as a downstream service, ACM will only handle the deployment of certificates to AWS-native services. The provisioning and management of certificates along with policy enforcement is done through upstream certificate service established by the InfoSec team. It will allow the Platform team to establish ACM as a single source of certificates in Azure and cater to different Development & Deployment teams using AWS-native services.

Security Strategy for HashiCorp Vault

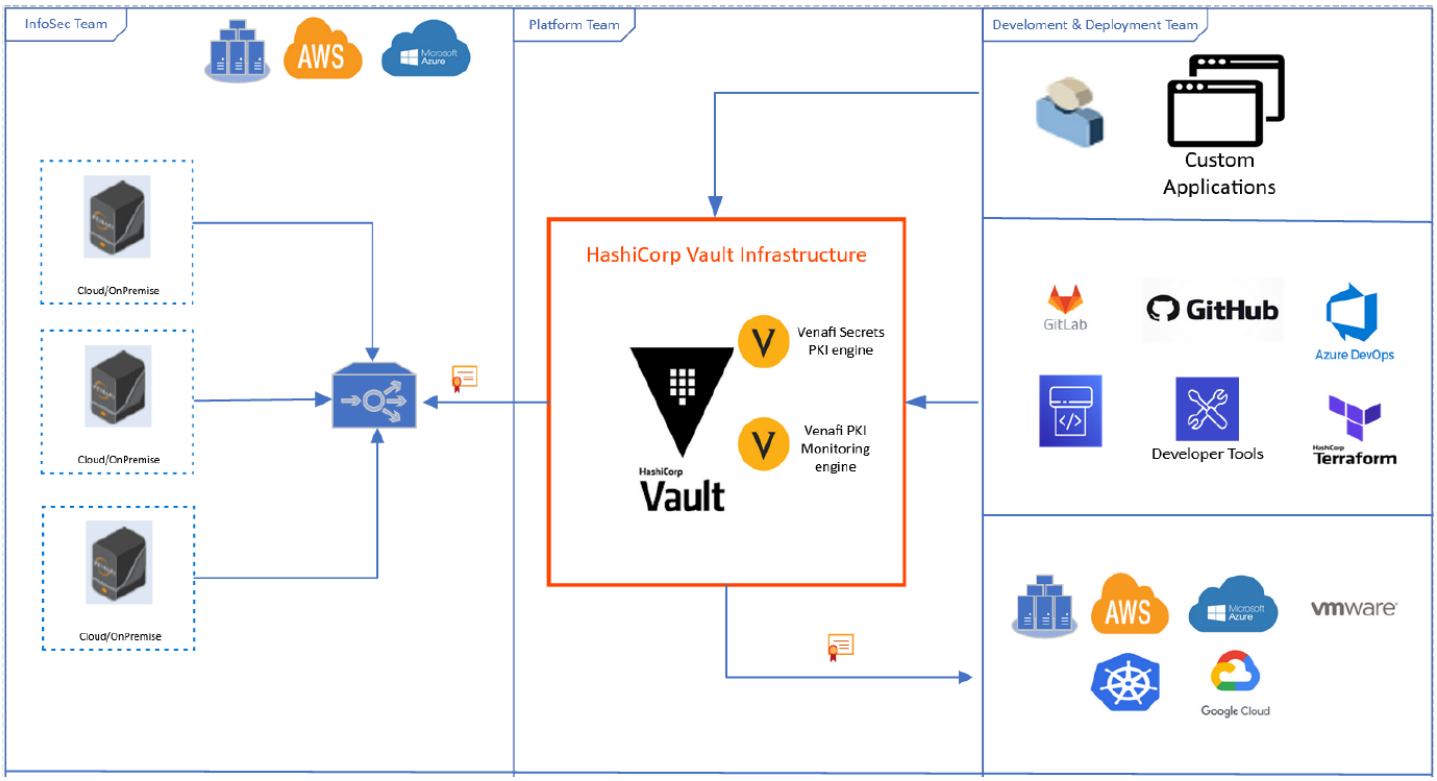

Figure 4: Strategy for HashiCorp Valut using a pull integration model

The above diagram shows a depiction of the proposed strategy for HashiCorp Vault using pull integration model for securely getting policy compliant certificates from upstream certificate service. Pull integration model is suggested as vault supports integrations with diverse infrastructure, tools and applications. InfoSec team deploys TPP as an upstream certificate service by proactively enforcing security policies, configuring notifications, reporting and assigning ownership to Platform and Development & Deployment teams. Platform teams can use 2 integrations with HashiCorp Vault to establish downstream certificate services. Venafi's Secrets PKI engine supports requesting certificate from TPP and allow for enforcement of policies. Venafi's PKI monitoring engine supports only policy enforcement through TPP, but the actual certificate is generated by HashiCorp Vault itself. Development & Deployment teams can use integration methods provided by vault itself to request certificates from downstream certificate service.

Security Strategy for Kubernetes

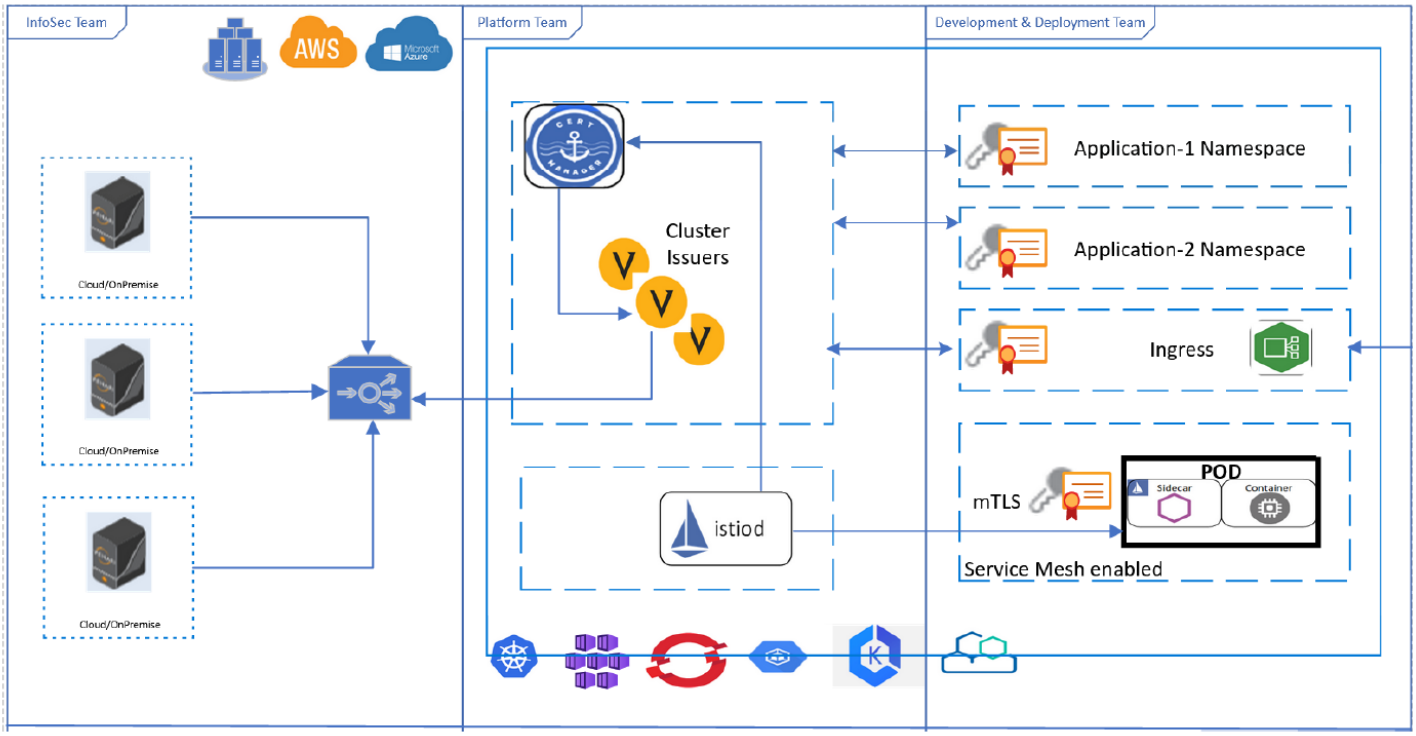

Figure 5: Security Strategy for Kubernetes using a pull integration model

The above diagram shows a depiction of the proposed strategy for Kubernetes (k8s) using pull integration model for securely getting policy compliant certificates from upstream certificate service. The downstream certificate service is implemented using cert-manager in k8s. Cert-Manager integrates with Venafi's TPP to provide certificates for Ingress, service mesh data plane and individual microservices. InfoSec team deploys TPP as an upstream certificate service by proactively enforcing security policies, configuring notifications, reporting and assigning ownership to Platform and Development & Deployment teams. Platform team configures and maintains cert-manager to use the TPP for all certificate requests and provide k8s native certificate templates for Development & Deployment teams, who will later the k8s native certificate templates (in YAML) to request certificate for microservices or ingress.

Security Strategy for IaC (Terraform/Ansible)

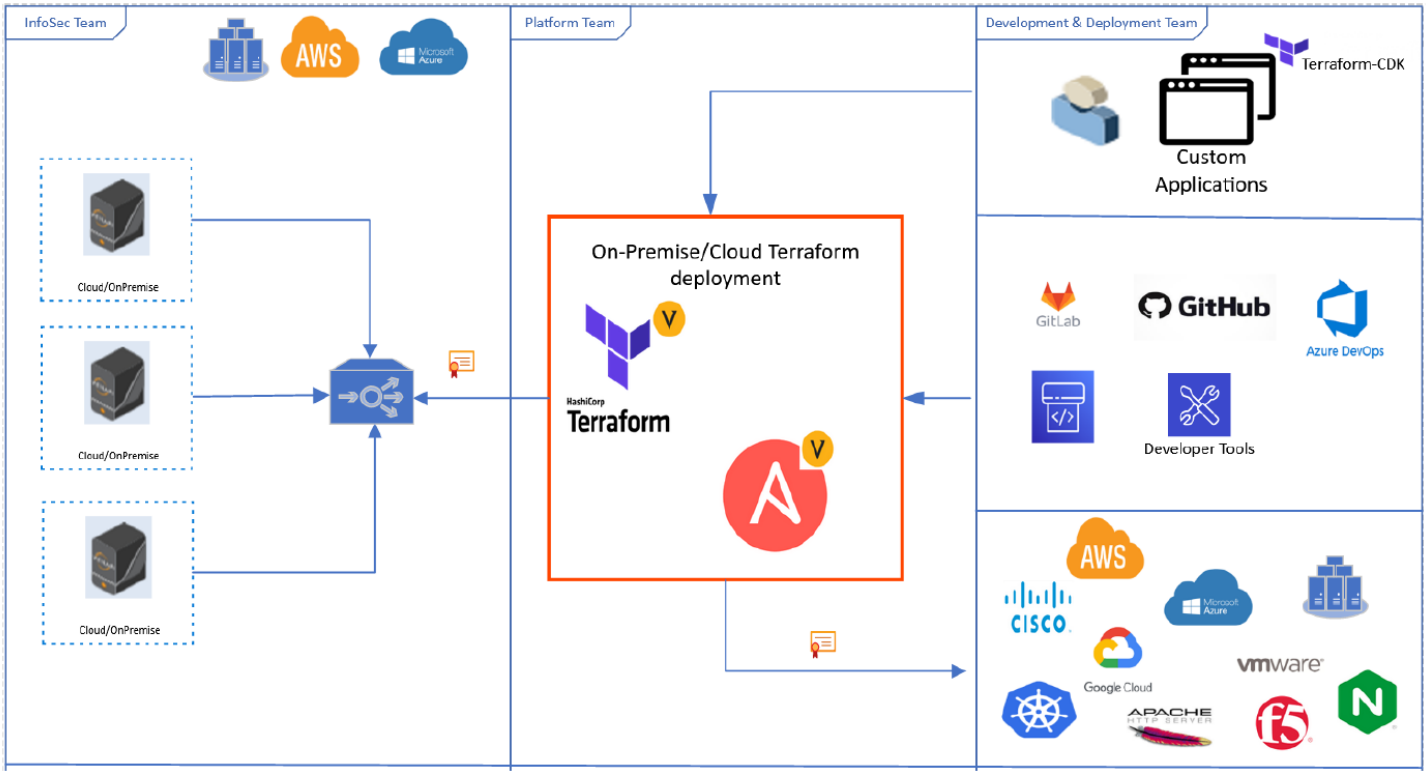

Figure 6: A strategy for infrastructure and configuration management platforms using a pull integration model

The above diagram shows a depiction of the proposed strategy for infrastructure and configuration management platforms using pull integration model for securely getting policy compliant certificates from upstream certificate service. Pull integration model is suggested as platforms like Terraform and Ansible support a wide spectrum of platforms, network devices, and services ranging from traditional to modern IT infrastructure. The downstream certificate service is implemented by using native integrations like Venafi's Terraform provider or Venafi ansible role. These integrations can be used to retrieve certificates from Venafi's TPP. InfoSec team deploys TPP as an upstream certificate service by proactively enforcing security policies, configuring notifications, reporting and assigning ownership to Platform and Development & Deployment teams. Platform team configures and maintains Venafi's Terraform provider or Venafi ansible role for all certificate requests and provide Terraform or Ansible native certificate modules or certificate roles for Development & Deployment teams, who will later them to request certificates.

Summary

Managing machine identities is a shared security concern between InfoSec, Platform and Development/Deployment teams. This paper introduced a unifying strategy to help address security requirements of all stakeholders in an enterprise. It also outlined implementation of the strategy with services commonly used in DevOps and Cloud environments using different integration models. The strategy will help remove barriers and act as bridge in understanding security requirements of different stakeholders, and help accelerate adoption of machine identities across the enterprise in organize and secure manner.