Google Cloud Whitepaper
August 2020

# Google Cloud security foundations guide

Google Cloud

# I. Security Foundations Blueprint

**Disclaimer:** *The content contained in this document is correct as of August 2020. This whitepaper represents the status quo as of the time it was written. Google Cloud's products, security policies, and systems might change going forward as we continually improve protection for our users.*

## Introduction

Cloud adoption continues to accelerate across enterprises, with more businesses moving from investigating the use of public cloud infrastructure to actually delivering production services to their customers through public clouds. Security in public clouds differs intrinsically from customer-owned infrastructure because there is shared responsibility for security between the customer and the cloud provider. Figure 1.1 shows a matrix of shared security responsibility for workloads in the cloud.

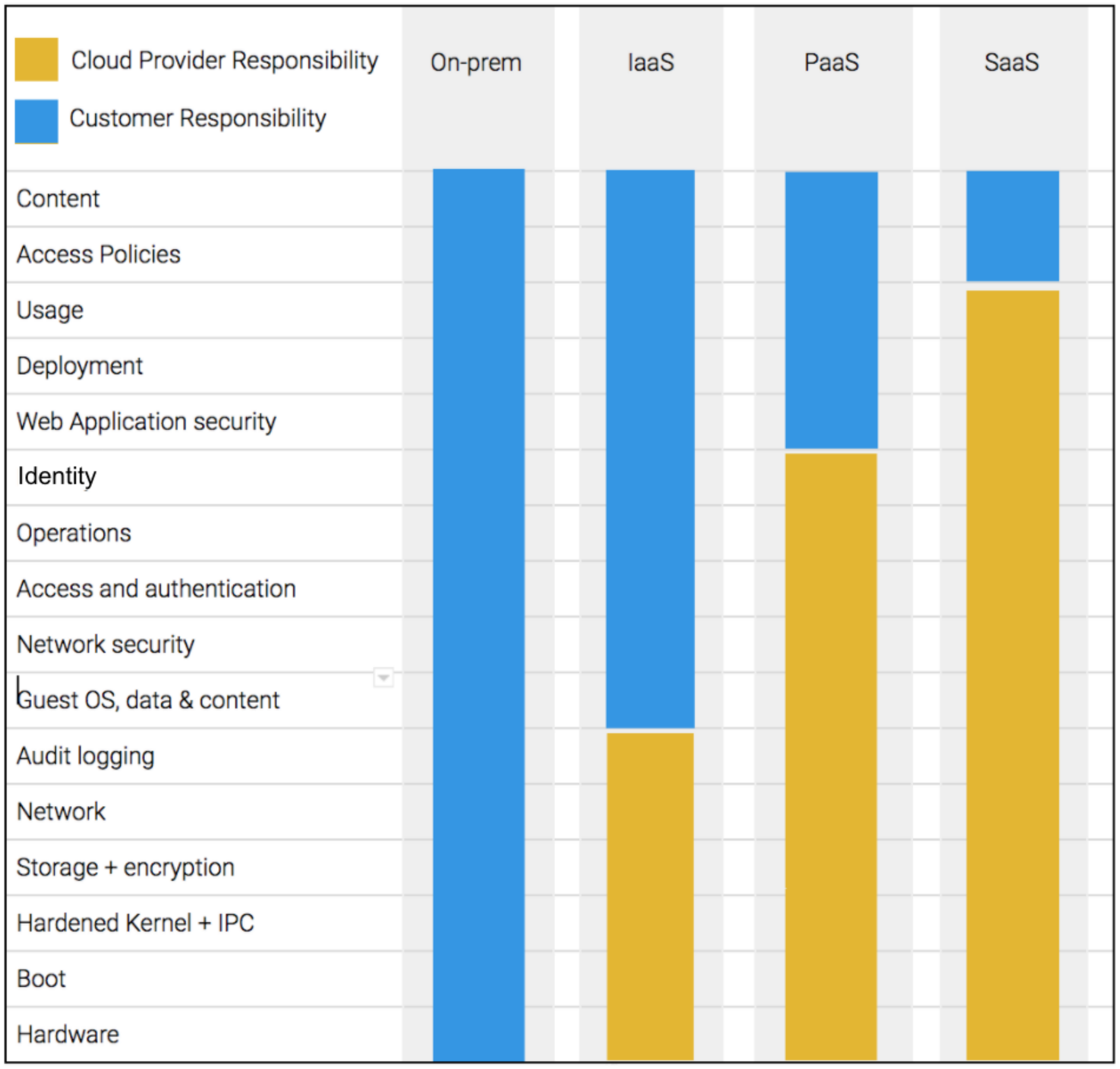


Figure 1.1 Shared security responsibilities

Google Cloud product and service offerings range from classic platform as a service (PaaS), to infrastructure as a service (IaaS), to software as a service (SaaS). The catalog of Google Cloud offerings continues to grow rapidly, making it important for you to have a **reliable, secure foundation to start with** and then to leverage service-specific security guidance. As shown in Figure 1.1, the boundaries of responsibility between you and the cloud provider change based on the services you've selected. As part of their shared responsibility for security, public cloud providers should enable you to start with a solid, secure foundation. Providers should then empower and make it easy for you to understand and execute your part of the shared responsibility model.

## Starting off with security in mind

Cloud Security is different from on-premises security because of the combination of the following:

- Differences in security primitives, visibility, and control points within the infrastructure, products and services.
- New cloud-native development methodologies like containerization and DevSecOps.
- The continued velocity and variety of new cloud products and services and how they can be consumed.
- Cultural shifts in how organizations deploy, manage, and operate systems.

You have many decisions to make when setting up your Cloud deployment. You might need help deploying workloads with secure defaults simply, quickly, and effectively in order to accelerate your ability to deliver business impact and value to your customers. But you might not have the time to build the new skills necessary to cope with the differences and new challenges of a cloud transition. Therefore, you can often benefit from curated and opinionated guidance for both a secure foundational starting point and for customization to match your specific needs.

Here's good news: we can help you build on Google Cloud more quickly, effectively, and securely by providing you with **security blueprints**, which add an important new set of layers in the hierarchy of security on Google Cloud. All in all, there are now four main parts, starting with core Google Cloud infrastructure as the base. The three subsequent layers are Google Cloud products and services; security foundations blueprints; and blueprints for security posture, workloads, and applications. Each builds on top of the previous layers.

### Core Google Cloud infrastructure

Google has a global-scale technical infrastructure that's designed to provide security through the entire information processing lifecycle at Google. This infrastructure provides secure deployment of services, secure storage of data with end-user privacy safeguards, secure communications between services, secure and private communication with customers over the internet, and safe operation by administrators. Google Cloud services are built on top of this infrastructure and are operated consistently in accordance with the cloud-native security principles that are shared in our BeyondProd whitepaper.

## Google products and services

Google Cloud products are designed with security in mind based on the approach and principles we outlined in the BeyondProd paper. They include built-in security features to enable access control, segmentation, and data protection. In addition, Google Cloud builds specific security products to help you meet your policy requirements and help protect your critical assets with our security products and capabilities.

### Security foundations blueprints

The goal of **this security foundations blueprint** is to provide you with curated, opinionated guidance and accompanying automation that helps you build a secure starting point for your Google Cloud deployment. This security foundations blueprint covers the following:

- Google Cloud organization structure
- Authentication and authorization
- Resource hierarchy and deployment
- Networking (segmentation and security)
- Logging
- Detective controls
- Billing setup

### Security posture, workload, and applications blueprints

On top of strong security foundations, we provide additional blueprints for security posture, workload, and applications to give you curated, opinionated guidance to help design, build, and operate workloads and applications. Examples include our PCI on GKE blueprint and Healthcare Data Protection Toolkit.

Figure 1.2 shows the layers of security that you can build on when you use Google Cloud and follow the guidance and templates provided by the security blueprints.

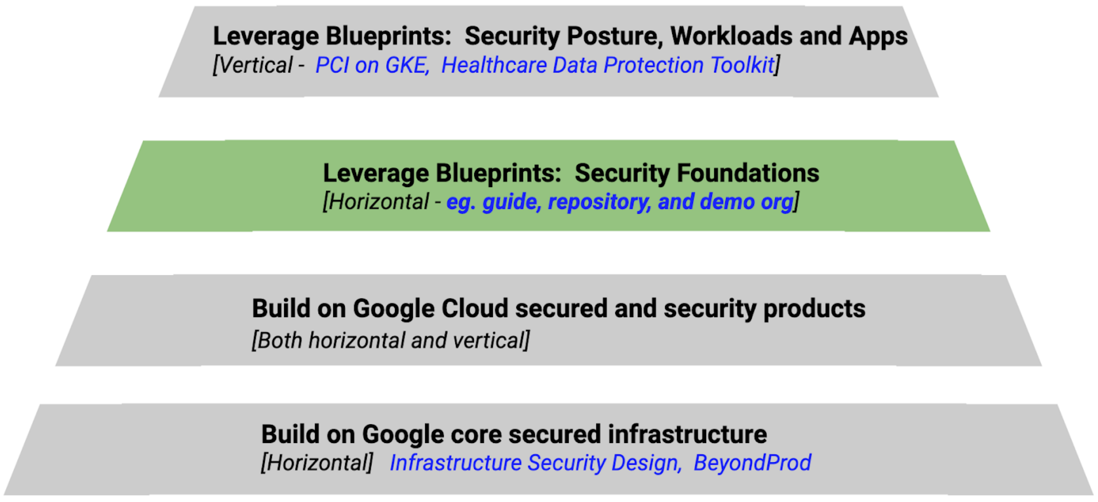

Leverage Blueprints: Security Posture, Workloads and Apps
[Vertical - PCI on GKE, Healthcare Data Protection Toolkit]
Leverage Blueprints: Security Foundations
[Horizontal - eg. guide, repository, and demo org]
Build on Google Cloud secured and security products
[Both horizontal and vertical]
Build on Google core secured infrastructure
[Horizontal] Infrastructure Security Design, BeyondProd

**Figure 1.2** Google Cloud hierarchy of security

## Beginning with a security foundations blueprint

This guide provides our opinionated security foundations blueprint and captures a step-by-step view of how to configure and deploy your Google Cloud estate. This document can provide a good reference and starting point because we highlight key topics to consider. In each topic, we provide background and discussion of why we made each of our choices. In addition to the step-by-step guide, this security foundations blueprint has an accompanying Terraform automation repository and an example demonstration Google organization so you can learn from and experiment with an environment configured according to the blueprint.

The balance of this document is organized into sections that cover the following:

- This introduction
- The foundation security model
- Foundation design
- The example.com sample that expresses the opinionated organization structure
- Resource deployment
- Authentication and authorization
- Networking
- Secrets management
- Logging
- Detective controls
- General security guidance

# II. Step-by-step guide

## 1. Introduction

This section provides a guide to support building a secure foundation for Google Cloud deployments, using the example.com organization as a reference enterprise for this architecture. As shown in Figure 2.1, you can help achieve security for example.com by providing an opinionated foundation, which provides a basis for workloads to run securely in the cloud. You can find scripts, code, and other deployment artifacts for the example.com organization in the `terraform-example-foundation` GitHub repository.

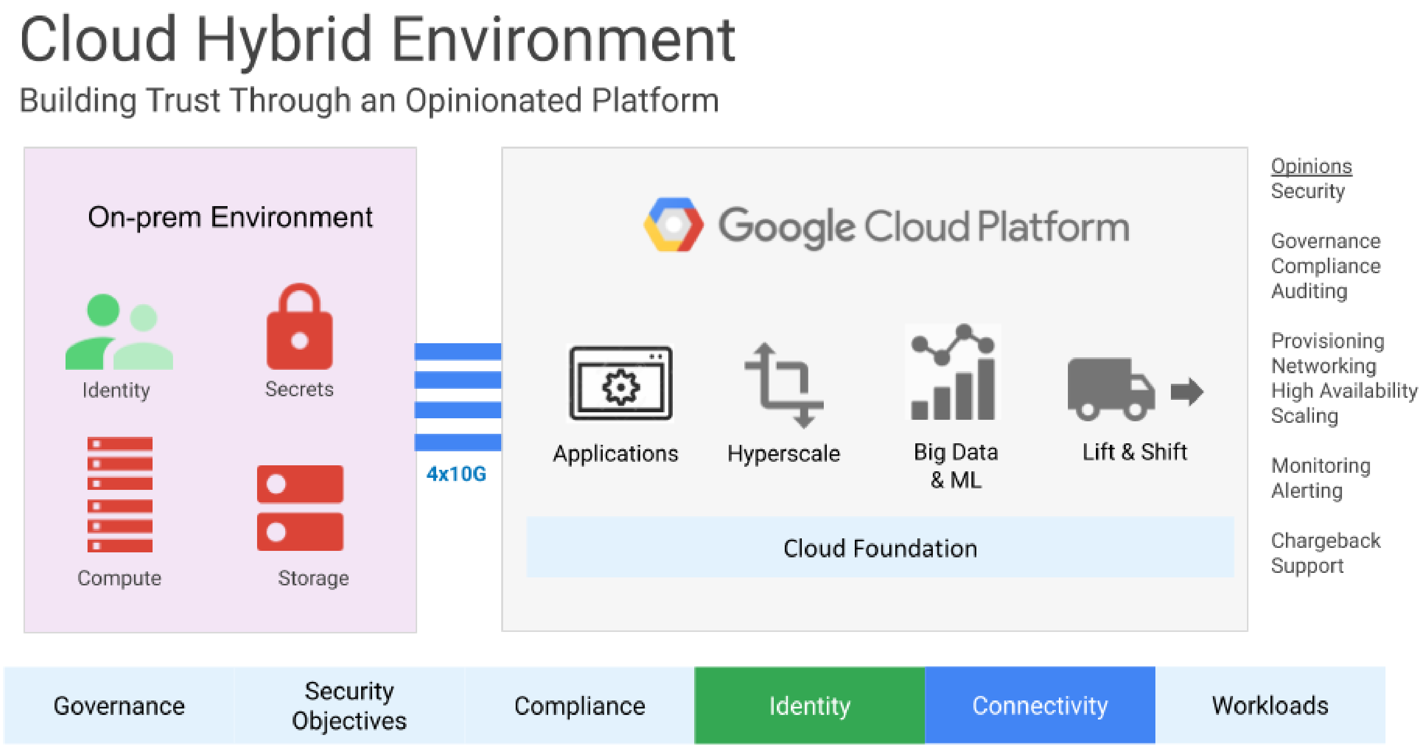


**Figure 2.1** Google Cloud opinionated security foundation

## 2. Google Cloud foundation security model

Foundation security in Google Cloud is enabled through a combination of preventative controls and detective controls. Preventative controls are realized through policy and architecture. Policy is defined as a series of programmatic constraints that protect the organization from threats. Architecture is defined as how infrastructure constructs can be used to protect the organization.

Detective controls are defined as monitoring capabilities that look for anomalous or malicious behavior within the organization. Figure 2.2 depicts how these three pillars of the security model come together to form the example.com reference architecture.

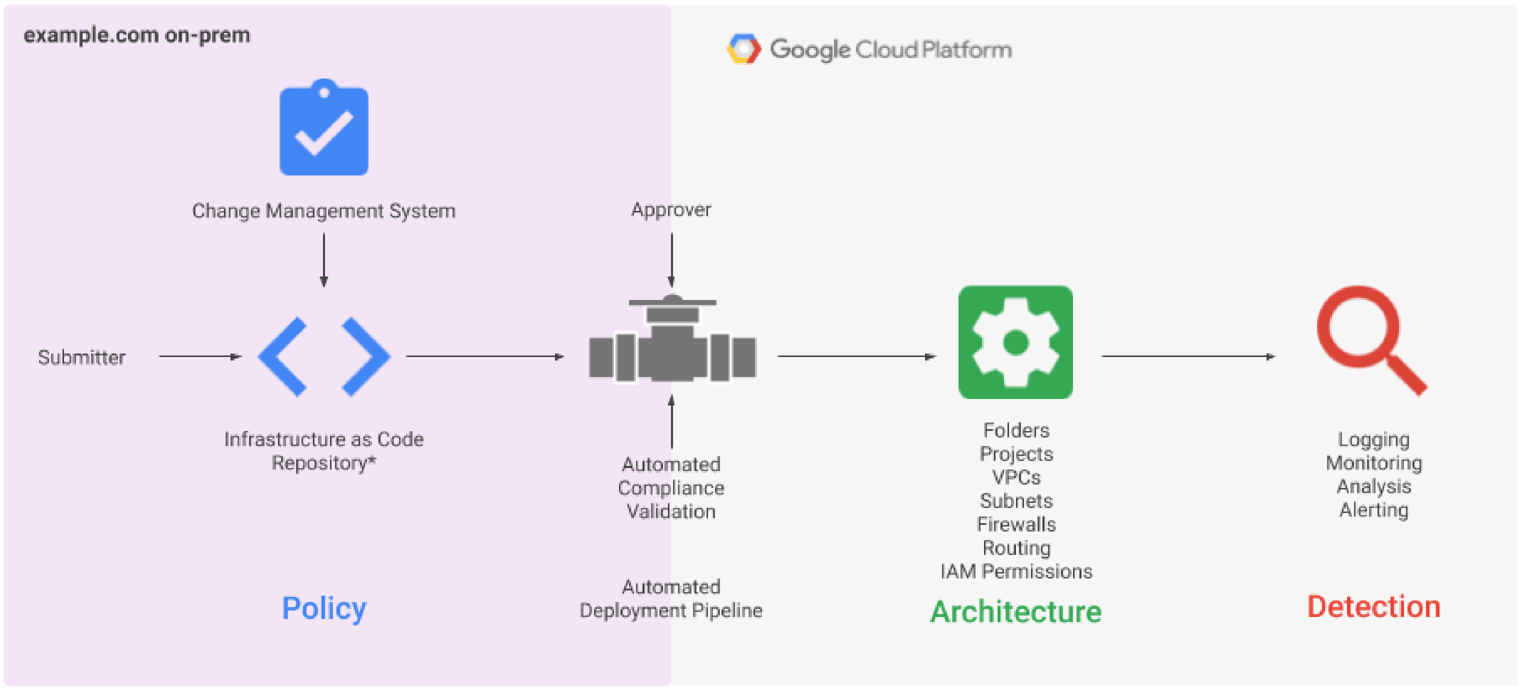


**Figure 2.2** The example.com security model

Deployment best practices, as detailed in Section 5, are implemented through codifying the Google Cloud infrastructure into Terraform modules, putting the modules under source code control, and deploying those modules through an automated pipeline that has policy validation and approval stages. The architectural constructs, as detailed in Section 6 through Section 9, encompass the folder and project structure of the Google Cloud organization, connectivity, networking structure, firewalls, and identity management. Detective controls are detailed in Section 10 and explain Google Cloud's platform capabilities to detect vulnerabilities, misconfiguration, and malicious behavior in order to enable security posture management. Section 10 also covers how to leverage Google Cloud's native capabilities in security analytics, as well as how to integrate Google Cloud capabilities with third-party SIEM tools.

# 3. Google Cloud foundation design

## 3.1) Key architectural decisions

The example.com reference architecture that's described in this document is based on certain architectural decisions; if you need to make changes to those assumptions, the example.com architecture also needs to be modified. The following table (Table 2.1) lists architectural decisions.

| Decision area | Decision |
|---|---|
| Organization structure | A single organization is used for all environments and services for example.com to manage all resources and policies in one place. |
| | The folder hierarchy has a single layer, consisting of bootstrap, common, production, non-production, and development folders to allow for segregation of policies, privileges, and access. |
| | Google Cloud organizational policies are used to augment the organization's security posture by allowing you to define resource configuration constraints that apply consistently across all projects. |
| Resource deployment | Foundation resources are deployed through a deployment pipeline to enable automated and standardized review, approval, and rollback. |
| | Terraform is used as the infrastructure as code (IaC) tool. |
| | An on-premises Git repository is used as a source repository for the Terraform modules. |
| | The deployment pipeline’s actions are initiated from the on-premises environment. |
| | An approval stage in the pipeline is needed to deploy resources. |
| | Workloads are deployed through separate pipelines and access patterns. |
| Authentication and authorization | Google Cloud federates with the example.com on-premises Active Directory identity management system. |
| | Single sign-on (SSO) is used to allow users to log into Google Cloud. |
| | A firecall process is used to provide elevated access to the Google Cloud environment. |
| | Groups are used for assigning Cloud IAM permissions. |
| Networking | Dedicated Interconnect connections are used to connect the on-premises environment with Google Cloud in a configuration that provides an SLA of 99.99%. |
| | Shared VPC is used as the primary networking topology to enable centralized management of firewall rules and connectivity. |
| | VPC Service Controls provide perimeter protection for services that store highly sensitive data to enable service-level segmentation. |

| | |
|---|---|
| | Access to Google Cloud APIs from the cloud and from on-premises is through private IP addresses. |
| | Address allocation for Google Cloud is a contiguous RFC 1918 /16 space. |
| | Cloud DNS is used in Google Cloud, and DNS forwarding is used to communicate with on-premises DNS servers. |
| | Tag-based firewall rules are used to control network traffic flows. |
| Secrets management | Google Secret Manager is used to store secrets. |
| | Separate Secret Manager instances are used to store organization-level and folder-level secrets. |
| Logging | Organization-level log sinks are used to aggregate logs. |
| | Logs are sent to Cloud Storage, to BigQuery, and to a SIEM through Cloud Pub/Sub. |
| Detective controls | An enterprise SIEM product integrates with Google Cloud Logging. |
| | Security Command Center Premium is enabled to detect infrastructure misconfigurations, vulnerabilities, and active threat behavior, and to share those findings with the enterprise SIEM tools. |
| | Logs in BigQuery are used to augment detection of anomalous behavior by Security Command Center Premium. |
| Billing | Billing alerts are used on a per-project basis with thresholds set at 50%, 75%, 90%, and 95%. |

**Table 2.1** Key architectural decisions

### 3.2) Pre-work

The reference architecture that's described in this document is based on the assumption that you've completed the following pre-work:

- ❏ You have set up Cloud Identity.
- ❏ You have validated your Google Cloud domain.
- ❏ You have established a billing account.
- ❏ You have reconciled conflict accounts.
- ❏ You have set up your Dedicated Interconnect connections.

For more information, see the Google Cloud onboarding documentation.

## 3.3) Naming conventions

We recommend that you have a standardized naming convention across various Google Cloud resources. Table 2.2 lays out conventions for creating names for elements in Google Cloud for the example.com reference architecture. Table 2.3 describes the field values that are used in the names.

**Note:** Fields listed in braces ({ }) are optional.

| Resource | Naming convention |
|---|---|
| Folder | `fldr-`*`environment`*<br><br>Example: `fldr-prod` |
| Project | `prj-`*`business-code`*`-`*`environment-code`*`{`*`-project-label`*`}-`*`unique-number`*<br><br>Example: `prj-acde-p-shared-base-43212` |
| VPC | `vpc-`*`environment-code`*`-`*`vpc-type`*`{-`*`vpc-label`*`}`<br><br>Example: `vpc-p-shared-base` |
| Subnet | `sb-`*`vpc-name`*`-`*`region`*`{-`*`subnet-label`*`}`<br><br>Example: `sb-p-shared-base-us-east1-net1` |
| Firewall | `fw-`*`vpc-name`*`-`*`priority`*`-`*`direction`*`-`*`action`*`-`*`src-label`*`-`*`dest-label`*`-`*`protocol`*`-`*`port`*`{-`*`firewall-label`*`}`<br><br>Example: `fw-p-shared-base-1000-i-a-all-all-tcp-80` |
| Cloud router | `cr-`*`vpc-name`*`-`*`region`*`{-`*`cloud-router-label`*`}`<br><br>Example: `cr-p-shared-base-us-east1-cr1` |
| Route | `rt-`*`vpc-name`*`-`*`priority`*`-`*`instance-tag`*`-`*`next-hop`*`-{`*`route-label`*`}`<br><br>Example: `rt-p-shared-base-1000-all-default-windows-activation` |
| Cloud Interconnect connection | `ic-`*`onprem-dc`*`-`*`colo`*<br><br>Example: `ic-dal-lga-zone1-1422` |
| Cloud Interconnect VLAN attachment | `vl-`*`ic-name`*`-`*`cr-name`*<br><br>Example: `vl-dal-da1-zone1-p-shared-base-us-east1-cr1` |
| Group | `grp-gcp-`*`group-label`*<br><br>Example: `grp-gcp-billing-admin` |

| | |
|---|---|
| Role | `rl-`*`function`*`{-`*`role-label`*`}`<br><br>Example: `rl-compute-admin` |
| Service account | `sa-{-`*`service-account-label`*`}`<br><br>Example: `sa-p-acde-shared-base-data-bkt-reader` |
| Storage bucket | `bkt-`*`project-name`*`{-`*`bucket-label`*`}`<br><br>Example: `bkt-p-acde-shared-base-data` |

**Table 2.2** Naming conventions for example.com

| Field | Description | Values |
|---|---|---|
| `environment` | A description of the folder-level resources within the Google Cloud organization. | `bootstrap`, `common`, `prod`, `nonprod`, `dev` |
| `environment-code` | A short form of the environment field. | `b`, `c`, `p`, n, d |
| `business-code` | A 4-character code that's used to associate a project with a business unit or group. | A uniquely identifiable 4-character code. For common projects that are not related to a business unit, `zzzz` is used. |
| `unique-number` | A globally unique identifier. | A 5-digit integer |
| `vpc-type` | The type of VPC network that's being established. | `shared`, `service`, `float`,nic, `peer` |
| `region` | The region that the resource is located in. | Any valid Google Cloud region. Short forms are used for some names and directions: Australia (au), North American (na), South America (sa), Europe (eu), southeast (se), and northeast (ne). |
| `priority` | A numerical value that specifies the priority of the Google Cloud route or firewall rule. | For details, see the documentation for firewall priorities and routing order. |
| `direction` | The direction of traffic relative to Google Cloud that the firewall applies. | `i` for ingress, `e` for egress |
| `action` | The action to take if a firewall rule matches. | `a` for allow, `d` for deny |
| `src-label` | The instance source label to which a firewall is applied. | `all` (indicating `0.0.0.0/0`), the source IP range, or source tags (list) |

| | | |
|---|---|---|
| `dest-label` | The instance destinations to which a firewall is applied. | `all`, the destination tags (list), or the service accounts (list) |
| `protocol` | The protocols to which a firewall is applied. | `all`, a single protocol, or a combination of protocols (`tcp`, `udp`, `tcpudp`, and so on) |
| `port` | The port or port range on which a firewall is applied. | A port number or port number range |
| `instance-tag` | The instances to which a route is applied. | instance tags |
| `next-hop` | The next-hop destination where the route will direct traffic, if applicable. | `default`, an instance ID, an IP address, a VPN tunnel name, oran internal load-balancer address |
| `onprem-dc` | The name of your data center to which an interconnect is connected. | Customer dependent |
| `colo` | The colocation facility name that the interconnect from the on-premises datacenter is peered with. | A valid Google Cloud colocation facility code |
| `function` | The name of the resource type that a custom role is associated with. | Resource dependent |
| `*name` | The name of a resource without its prefix. | Resource dependent |
| `*label` | A descriptive field to enhance the description of the resource. | Resource dependent |

**Table 2.3** Naming convention field values

## 4. The example.com Google Cloud organization structure

The foundation of creating deployments within Google Cloud is the organization, which you create by setting up Cloud Identity or G Suite. The Google Cloud organization provides a resource hierarchy that provides an ownership structure for resources and attachment points for policy and access control.

The resource hierarchy consists of folders, projects, and resources, and it defines the shape and use of Google Cloud services within an organization. Policies are inherited, and these policies cannot be altered by resource owners who are lower in the hierarchy. Folders and projects inherently provide you with secure-by-default isolation because all external network and access permissions are denied by default and therefore must either be explicitly enabled or enabled by inheritance from a higher level in the hierarchy.

Folder structure is dependent on the nature of the organization and the type of workload being deployed. Figure 2.3 shows the example.com reference architecture.

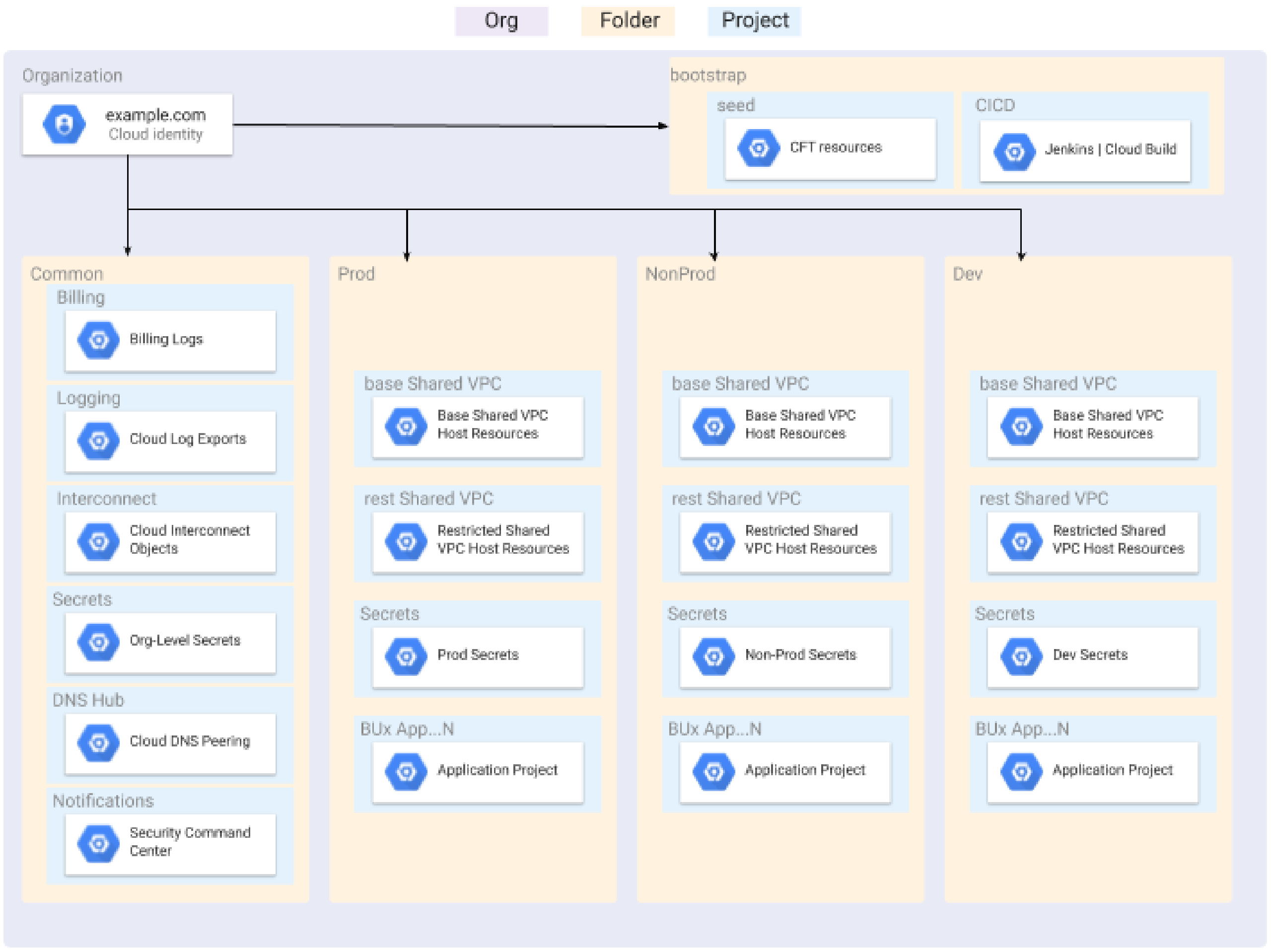


**Figure 2.3** The example.com organization structure

The architecture consists of five folders as described in Section 4.1 and of a number of projects that are detailed in Section 4.2. The organizational structure for example.com is codified in Terraform modules and deployed through the deployment pipeline as detailed in Section 5.

**Note:** The example.com folder structure provides a basis for a typical application-deployment architecture. However, for production scenarios, you should tailor your deployment strategy to your specific needs around resource isolation, geographic operating needs, business unit autonomy, data isolation, and access control. In addition, some of your workloads such as analytics might need production data in a non-production environment. This might lead you to create different environmental classifications, such as an analytics folder. You might also need to create multiple Google Cloud organizations to increase separation and isolation.

### 4.1) Folders

You use folders as a method of grouping projects into related groups. You can apply security policies and access permissions at the folder level; these policies and permissions are then inherited by resources within the folder structure.

The key concept of folders is that they separate projects into logical groupings; you can then apply consistent policy to the child projects at the folder level to ease administrative overhead. As shown in Table 2.4, the reference architecture used by example.com uses five folders. Three of them (production, non-production, and development) separate the organizations into different environments. These folders are deployed through the deployment pipeline detailed in Section 5.

| Folder | Description |
|---|---|
| `bootstrap` | Contains the seed and CICD projects that are used to deploy foundation components. |
| `common` | Contains projects with cloud resources used by the organization. |
| `production` | Contains projects with cloud resources that have been promoted into production. |
| `non-production` | Contains a replica of the production environment to let you test workloads before you put them into production. |
| `development` | Used as a development and sandbox environment. |

**Table 2.4** Folders used in the example.com reference architectures

### 4.2) Projects

Projects in Google Cloud are constructs that contain cloud resources. Each project inherits the policy of the parent folder or folders and of the Google Cloud organization. You can enable or disable APIs within each project to support the services that you need for that project. Each project is associated with a billing account, as explained in Section 5.3.4.

#### 4.2.1) Common folder and bootstrap folder projects

In the example.com architecture, a series of projects reside under the common folder and bootstrap folder that contain resources that are used across the example.com organization. These projects, detailed in Table 2.5, provide various enterprise functions and are created through the infrastructure deployment pipeline.

| Project | Description | More information |
|---|---|---|
| CICD | Houses the deployment pipeline that's used to build out the foundation components of the organization. This project is highly restricted. | Section 5 |
| seed | Contains the Terraform state of your infrastructure and Terraform service account to update the state. This project is highly restricted. | Section 5 |
| interconnect | Houses the Cloud Interconnect connections that provide connectivity between an enterprise's on-premises environment and Google Cloud. | Section 7 |
| DNS hub | Provides a central point of communication between an enterprise’s on-premises DNS system and Cloud DNS. | Section 7 |
| global secrets | Contains organization-level secrets. | Section 8 |
| logging | Provides a destination for log sink and detective controls. | Section 9 |
| Security Command Center | Provides a standalone project for managing Security Command Center alerting. | Section 10 |
| billing | Contains a BigQuery dataset with the organization's billing exports. | Section 11 |

**Table 2.5** Projects in the common folder

### 4.2.2) Projects present in all environment folders

Under each environment folder (production, non-production, and development) in the example.com reference architecture, a series of common projects are created, as detailed in Table 2.6. These projects provide resource connectivity to the on-premises environment and to a deployment environment for workload-specific resources.

| Project type | Description |
|---|---|
| Base Shared VPC host project | Provides the project for the Shared VPC network that hosts the organization's workloads that do not require VPC Service Controls. |
| Restricted Shared VPC host project | Provides the project for the Shared VPC network that will host the organization's workloads that run within a perimeter controlled by VPC Service Controls. |
| secrets | Contains folder-level secrets. |
| application projects | Involve projects where application resources are deployed. Project deployment patterns are described in Section 7.1.1 and consist of Shared VPC service project patterns, floating project patterns, peered project patterns, and dual-NIC project patterns. |

**Table 2.6** Environment folder project types

## 4.3) Organization policy setup

You can use the Organization Policy Service to apply policies to a Google Cloud organization. The organizational policies are inherited by all child resources of the resource node to which the policy is set, unless an explicit override is set on a child resource. Organization policy constraints in the example.com reference architecture are defined in Terraform modules and deployed through the deployment pipeline. Table 2.7 provides a list and description of the policies that are set as part of the example.com foundation.

| Policy constraint | Description | Recommended value |
|---|---|---|
| `compute.disableNestedVirtualization` | (boolean) When `true`, disables hardware-accelerated nested virtualization for all Compute Engine VMs. | `true` |
| `compute.disableSerialPortAccess` | (boolean) When `true`, disables serial port access to Compute Engine VMs. | `true` |
| `compute.disableGuestAttributesAccess` | (boolean) When `true`, disables Compute Engine API access to the guest attributes of Compute Engine VMs. | `true` |

| | | |
|---|---|---|
| compute.vmExternalIpAccess | (list) Defines the set of Compute Engine VM instances that are allowed to use external IP addresses. | `deny all=true` |
| compute.skipDefaultNetworkCreation | (boolean) When `true`, causes Google Cloud to skip the creation of the default network and related resources during Google Cloud project resource creation. | `true` |
| compute.restrictXpnProjectLienRemoval | (boolean) When `true`, restricts the set of users that can remove a Shared VPC project lien. | `true` |
| sql.restrictPublicIp | (boolean) When `true`, restricts public IP addresses on Cloud SQL instances. | `true` |
| iam.allowedPolicyMemberDomains | (list) Defines the set of members that can be added to Cloud IAM policies. The allowed/denied list must specify one or more Cloud Identity or G Suite customer IDs.<br><br>Note that domain restricted sharing can interfere with some Google Cloud services, and you might need to provide exceptions for some Google Cloud services. | *your-cloud-identity-id* |
| iam.disableServiceAccountKeyCreation | (boolean) When `true`, disables the creation of service account external keys. | `true` |
| storage.uniformBucketLevelAccess | (boolean) When `true`, requires buckets to use uniform IAM-based bucket-level access. | `true` |

**Table 2.7** Organizational policies in example.com

**Note:** You might need to modify organization policy constraints to accommodate certain application types or workflows—for example, configuring new log exports, purposefully creating public-facing buckets, or giving a VM an external IP address for internet access.

### 4.4) Additional policy controls

In addition to the organizational policy constraints detailed in the previous section, Google Cloud has additional policies that you can apply to an organization to tighten the organization's security posture. Table 2.8 lists the additional policy controls that are used in the example.com reference architecture.

| Control | Description | Management |
|---|---|---|
| Limit session and gcloud timeouts | You can change the session timeout for Google Cloud sessions (including the `gcloud` tool) to durations as low as 1 hour. | G Suite or Cloud Identity Admin Console |

| | | |
|---|---|---|
| Disable Cloud Shell | Google Cloud provides a native Cloud Shell interface for working with the cloud environment. You can disable this shell. | Admin Console |
| Use phishing-resistant security keys | You can enable phishing-resistant security keys as a second-factor authentication (2FA) method that helps provide protection against automated bots, bulk phishing, and targeted attacks. The keys use public key cryptography to verify a user's identity, and they use the URL of the login page to help make sure that an attacker can't access a user account even if the user is tricked into providing a username and password. | G Suite or Cloud Identity Admin Console. If you're federating identities from an external system, you might need to configure this in your identity provider, or add an additional layer of protection on top of your existing SSO settings directly within Cloud Identity. |
| Enable access transparency | Access transparency provides you with logs that capture the actions that Google personnel take when accessing your content. You must be on Gold, Platinum, Enterprise, or Premium support plans. | Contact Google Cloud Support |
| Enable access approval | Access approval enables you to explicitly approve access to your data or configurations on Google Cloud before Google Cloud support or engineering can access the data. | Google Cloud Console |
| Restrict data storage locations | You can limit the physical location of a new resource in Google Cloud by using resource location constraints. | Same mechanism as organization policy controls detailed in Section 4.3. |

**Table 2.8** Additional example.com policy controls

**Note:** There is an important distinction between allowing any 2FA scheme, most of which don't resist phishing, and requiring a phishing-resistant 2FA mechanism such as Titan Security Keys or other keys based on the phishing-resistant FIDO U2F or CTAP1 standards. Enterprises can enforce 2-step verification, including security key verification, on accounts gradually, which allows migration. Enforcement can then be enabled for more privileged users first and expanded as practical.

## 5. Resource deployment

Resources in example.com reference architecture can generally be grouped into one of two categories: foundation or workload components. Foundation resources need to be tightly secured, governed, and audited to avoid exposing the enterprise to any security or compliance risks. Foundation resources include resources in the hierarchy such as organizational policies, folders, projects, APIs, identities (for example, service accounts), role bindings, custom role definitions, Shared VPC networks, subnets, routes (dynamic and static), firewall rules, and Dedicated Interconnect connections.

Resources within Google Cloud can be deployed from the Google Cloud Console web user interface, using the gcloud command-line tool, using API calls directly, or using an infrastructure as code (IaC) tool. For creating the foundation elements of your enterprise's environment, you should minimize the amount of manual configuration so that you limit the possibility of human error. In the example.com reference architecture, Terraform is used for IaC, deployed through a pipeline that's implemented using Jenkins.

The combination of Terraform and Jenkins allows the foundation elements of example.com to be deployed in a consistent and controllable manner. The consistency and controllability of this approach helps enable governance and policy controls across the example.com Google Cloud environment. The example.com reference pipeline architecture is designed to be consistent with most enterprise's security controls.

### 5.1) CICD and seed projects

The example.com organization leverages functionality from the Cloud Foundation Toolkit to create the basic resources necessary to stand up an IaC environment within Google Cloud. This process creates a bootstrap folder at the root of the organization that contains a CICD project and a seed Terraform project.

The CICD project is a tightly controlled project within the organization hierarchy that's used to host the Jenkins deployment pipeline. It also hosts a service account that's used to run the Jenkins agents' Compute Engine instances. The Jenkins service account can impersonate the Terraform service account, which is located in the seed project, and which has permissions to deploy the foundation structures within the organization. The CICD project is created through a scripted process, and it has direct connectivity to the on-premises environment, separate from the connectivity described in Section 7.2.

The seed project contains the Terraform state of the foundation infrastructure, a highly privileged service account that's able to create new infrastructure, and the encryption configuration to protect that state. When the CI/CD pipeline runs, it impersonates this service account. The reason for having independent CICD and Terraform seed projects is due to separation of concerns. While Terraform is used as the IaC tool and has its own requirements, the deployment of that IaC is the responsibility of the CI/CD pipeline.

**Note:** The example.com reference architecture describes using Jenkins for the deployment pipeline. If you want to use a Google Cloud native service, the code repository also incorporates a Cloud Build-based pipeline.

### 5.2) Deployment pipeline architecture

Figure 2.4 shows the deployment pipeline for example.com. Terraform code that defines the example.com infrastructure is stored in an on-premises Git repository. Code changes to the main branch of the repository trigger a webhook that in turn triggers a deployment of the Terraform code into the example.com organization.

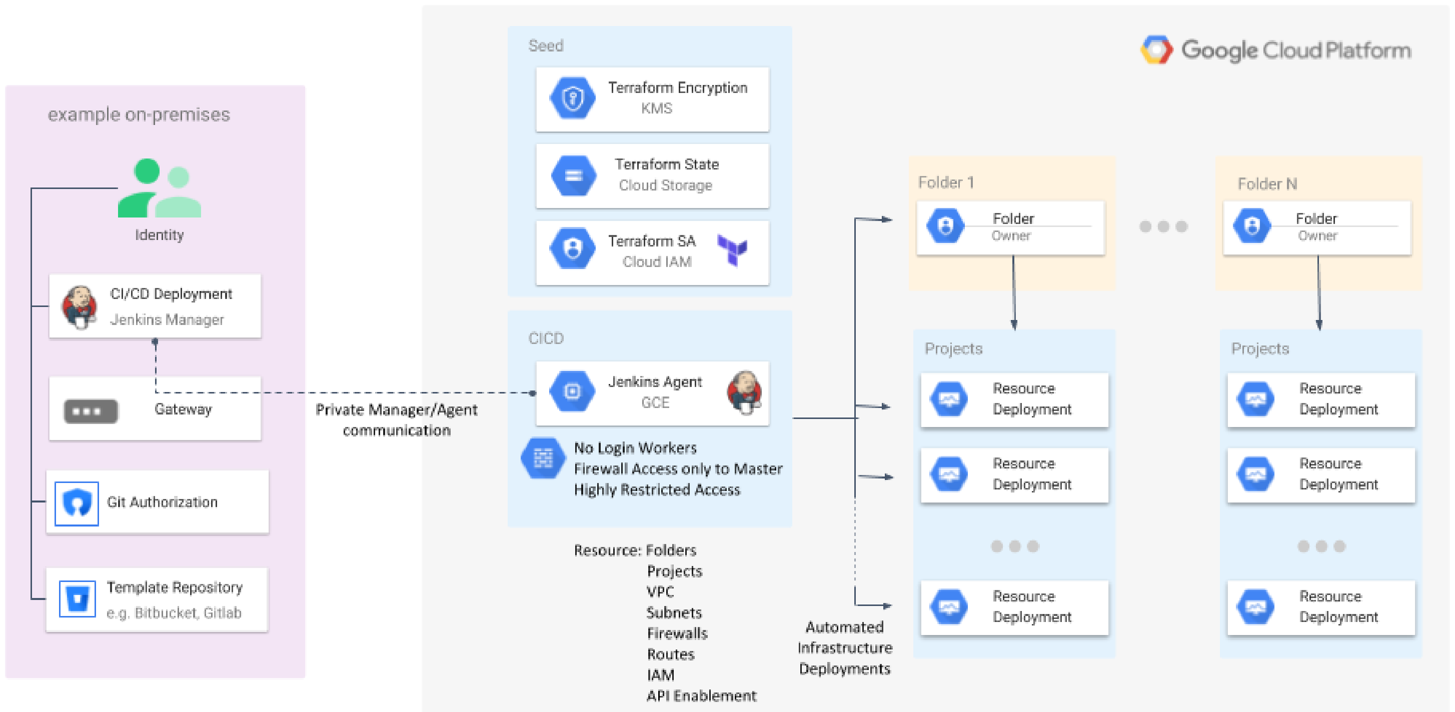


**Figure 2.4** The example.com foundation deployment pipeline

The Jenkins pipeline is implemented using a horizontal distributed-build architecture. In this model, a Jenkins manager (master) handles HTTP requests and manages the build environment, while the execution of builds is delegated to the Jenkins agents in the CICD project.

The Jenkins manager is hosted on-premises, co-located with the source-control management system, while the Jenkins agents are hosted in the Google Cloud environment. The Jenkins agent is a simple SSH server with Java and Docker installed on it—it needs to know only the manager's SSH public key. The Jenkins manager in turn acts as an SSH client. The manager uses the SSH private key to connect to the agent and deploys a Java executable JAR file that allows it to instruct the agent to execute the pipeline. The pipeline creates temporary containerized Terraform workers that deploy and modify the example.com infrastructure.

Table 2.9 details the security and governance controls that are integrated with the pipeline.

| Control | Description |
|---|---|
| Pull request (PR) | Code that's merged with the main branch needs to have an approved PR. |
| Policy checks | Policy checks are enforced by the pipeline against the Terraform code using Terraform Validator. |
| Deployment approval | An optional manual approval stage is included in the pipeline for deploying code. |

**Table 2.9** The example.com foundation pipeline security controls

Using this architecture in Google Cloud allows a clean authentication and authorization process for Google Cloud APIs. The Jenkins agent uses a custom service account that does not have permission to create new infrastructure and that instead impersonates the Terraform service account to deploy infrastructure. The Terraform service account in turn has the administrative roles that are needed in order to create organization policies, folders, projects, and other foundation components. Because these service accounts include very sensitive permissions, access to the CICD and seed projects where these agents run is highly restricted.

Running the agents in Compute Engine forgoes the requirement to download a service account's JSON key, which would be required if these agents ran on-premises or in any other environment outside of Google Cloud. Within the example.com organization, only the Jenkins agents and select firecall accounts have permissions to deploy foundation components.

**Note:** Policy should not be hard-coded directly into Terraform modules or hard-coded directly into deployment pipelines. Policy should be handled by services like the Terraform Validator or Open Policy Agent.

### 5.3) Project deployment

Projects in the example.com organization are deployed through the deployment pipeline. This section describes project attributes that are assigned when the project is created.

#### 5.3.1) Project labels

Project labels are key-value pairs that are included with the billing export and into Cloud Monitoring, enabling enhanced analysis on billing and resource usage. Table 2.10 details the project metadata that's added to each project in the example.com deployment.

| Label | Description |
|---|---|
| `business-code` | A 4-character code that describes which business unit owns the project. The code abcd is used for projects that are not explicitly tied to a business unit. |
| `billing-code` | A code that's used to provide chargeback information. |
| `primary-contact` | The primary email contact for the project. |
| `secondary-contact` | The secondary email contact for the project. |
| `environment` | A value that identifies the type of environment, such as `nonprod` or `prod`. |

**Table 2.10** Project labels for example.com

#### 5.3.2) IAM permissions

IAM permissions are defined and created on a per-project basis as part of project deployment.

#### 5.3.3) Google Cloud APIs

Google Cloud APIs are enabled on a per-project basis, and the pipeline has policy checks in place to ensure that only approved APIs can be enabled using an allow/deny list.

#### 5.3.4) Billing account

New projects are linked to the primary billing account. Chargeback to the appropriate business unit is enabled through the use of project labels, as described in Section 11.2.

#### 5.3.5) Networking

Project networking structures such as VPC networks, subnets, firewalls, and routes are enabled through the deployment pipeline.

### 5.3.6) Project editor

There are two project editors associated with a project. One is the custom service account that's used by the deployment pipeline in the seed project. The other project editor is a firecall account that can be used if automation breaks down or in an emergency, as described in Section 6.3.

## 5.4) Repository structure

The example.com code repository is distributed as a monorepo to make it easy for you to fork, copy, and use. However, the code has been constructed in such a way that each step in the code is executed through a separate Jenkins job and repository. The top-level folders and the contents of each folder are shown in Table 2.11. You can find the code for example.com in the `terraform-example-foundation` GitHub repository.

| Folder | Description | example.com components |
|---|---|---|
| `0-bootstrap` | This is where initial projects and IAM permissions are deployed for subsequent IaC stages (1-4). | `bootstrap` folder<br>● seed project<br>● CICD project<br>  ○ Jenkins pipeline<br>  ○ Terraform Validator |
| `1-org` | This is for organization-wide concerns such as policy, log exports, IAM, and so on. | organization policy<br>organization-wide IAM settings<br>`common` folder<br>● billing export project<br>● log export project<br>● interconnect project<br>● org-wide secrets project<br>● DNS project<br>● SCC notifications project |
| `2-environments` | This is for modular creation of new top-level environments, including required projects and the top-level folder. | `dev` folder<br>● base Shared VPC host projects<br>● restricted Shared VPC host projects<br>● environment secrets projects<br>● environment monitoring project<br>`nonprod` folder<br>● (same projects as dev folder)<br>`prod` folder<br>● (same projects as dev folder) |
| `3-networks` | This is for modular creation and management of VPC networks. | VPC networks<br>firewall rules<br>Cloud Routers<br>routes |
| `4-projects` | This is for creation of projects for different teams or business units, with an application workload focus. | application projects |

**Table 2.11** The example.com repository structure

## 5.5) The foundation pipeline and workloads

The pipeline architecture that's laid out in this section deploys the foundation layer of the Google Cloud organization. The pipeline is not intended for deploying higher-level services or applications. Furthermore, as shown in Figure 2.5, the access pattern to Google Cloud through deployment pipelines is only one potential access pattern. You might need to evaluate the access pattern and controls on the workload for each workload individually.

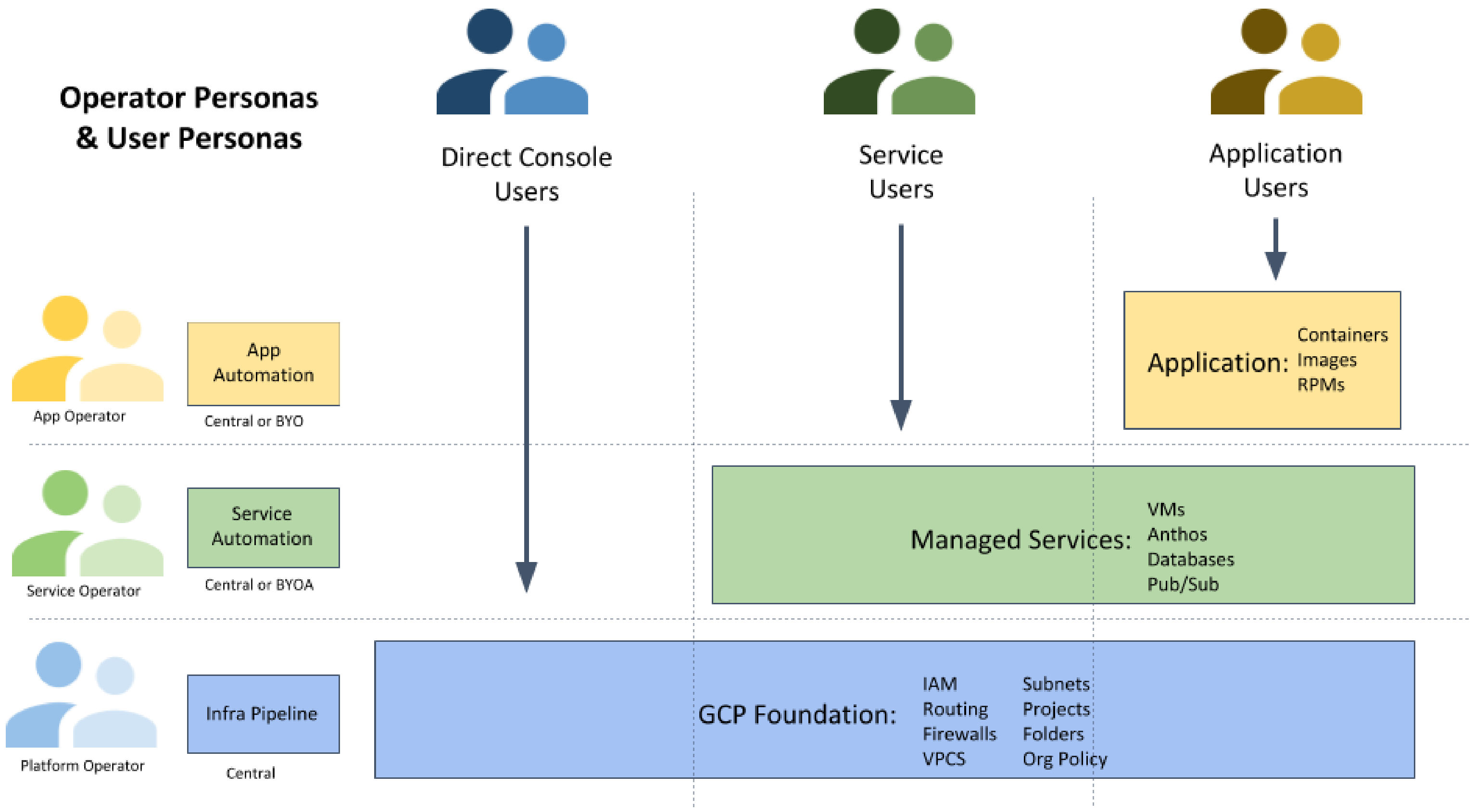

Operator Personas & User Personas
Direct Console Users
Service Users
Application Users
App Operator
App Automation
Central or BYO
Application: Containers Images RPMs
Service Operator
Service Automation
Central or BYOA
Managed Services: VMs Anthos Databases Pub/Sub
Platform Operator
Infra Pipeline
Central
GCP Foundation: IAM Routing Firewalls VPCS Subnets Projects Folders Org Policy

**Figure 2.5** Access patterns for example.com

# 6. Authentication and authorization

## 6.1) Cloud Identity, directory provisioning, and single sign-on

Google Cloud uses Google Cloud Identity for authentication and access management. Manually maintaining Google identities for each employee can add unnecessary management overhead when all employees already have an account in an existing identity store such as Active Directory. By federating user identities between Cloud Identity and Active Directory, you can automate the maintenance of Google identities and tie their lifecycle to existing identity management processes within your organization.

In the reference architecture, as shown in Figure 2.6, relevant users and groups are synchronized periodically from Active Directory to Cloud Identity using the Google Cloud Directory Sync tool. This process ensures that when you create a new user in Active Directory, that user can be referenced in Google Cloud. This process also ensures that if a user account is deleted or suspended, those changes are propagated. Provisioning works one way, which means changes in Active Directory are replicated to Cloud Identity but not the other way around. The sync process does not include passwords by default; in a federated setup, Active Directory remains the only system that manages these credentials.

**Note**: During initial setup, the Directory Sync process must be authenticated interactively using a 3-legged OAuth workflow. We strongly recommend that you create a dedicated account for this sync, as opposed to using an employee's admin account, because the sync will fail if the account performing the sync no longer exists or does not have proper privileges. The sync account requires the groups admin and user management Admin administrator roles within the Cloud Identity tenant, and you should protect access to the account as you would to any other highly privileged account. For additional best practices for using Directory Sync, see the documentation.

Single sign-on (SSO) is used in the example.com reference architecture for user authentication. During sign on, Google Cloud delegates authentication to Active Directory Federation Services by using the Security Assertion Markup Language (SAML) protocol. This delegation ensures that only Active Directory manages user credentials and that any applicable policies or multi-factor authentication (MFA) mechanisms are being enforced. For a sign-on to succeed, the user must be provisioned in both Active Directory and Cloud Identity.

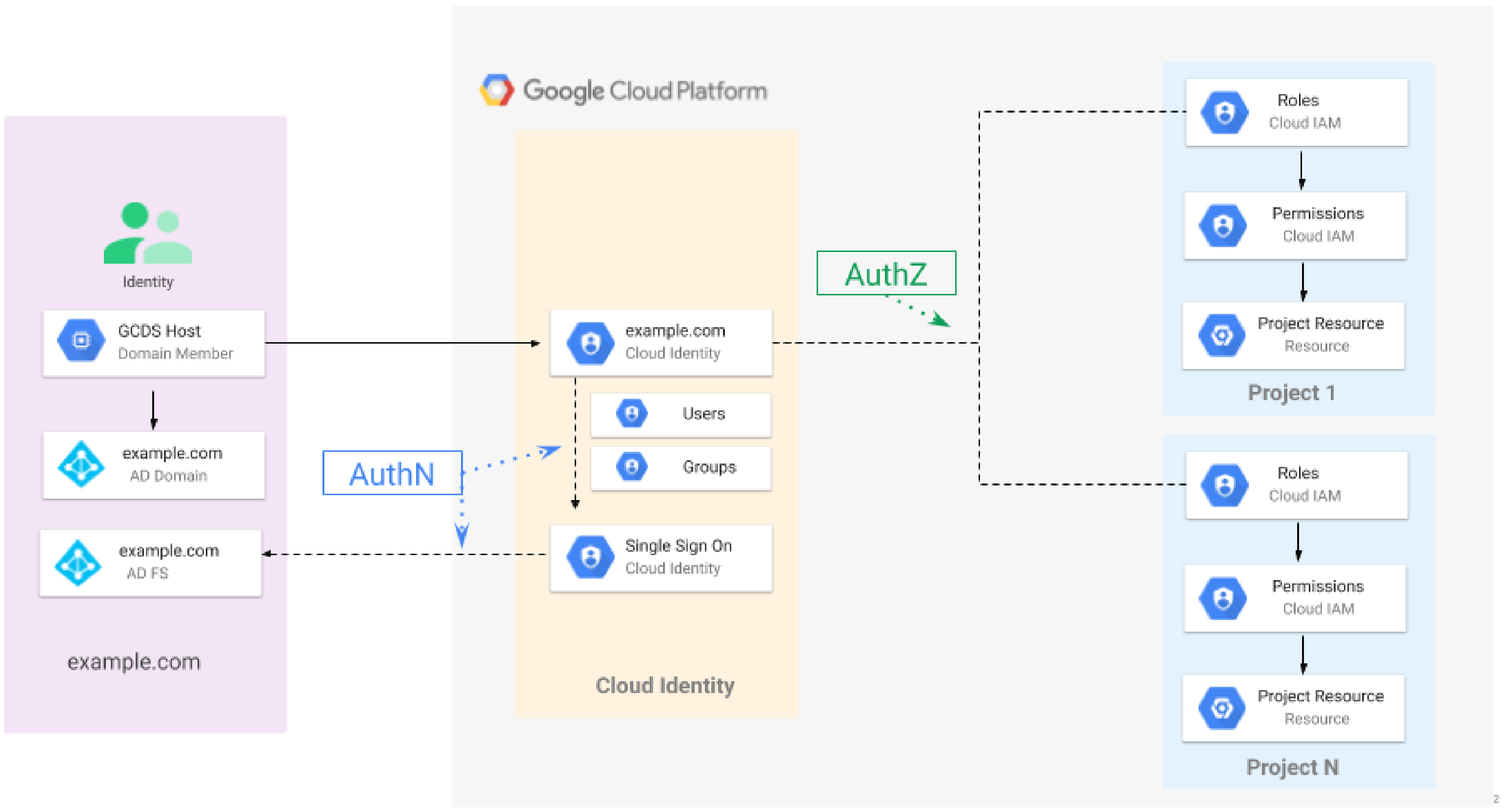


**Figure 2.6** The example.com identity structure

In the example.com reference architecture, the on-premises Active Directory system is used as the source of truth for users, groups, and SSO. Google Cloud Directory Sync is installed on a domain-joined host on-premises (either Windows or Linux). A scheduled job (using Task Scheduler or cron) is used to synchronize users and groups into Cloud Identity on an hourly basis. To control which users are synced to Cloud Identity, an attribute is added to the users in Active Directory, and Directory Sync filters on that attribute. In the example.com domain, Google Cloud-specific groups are prefixed with `grp-gcp-` (for example, `grp-gcp-billing-viewer` or `grp-gcp-security-reviewer`) and a filter is applied to sync only those groups. The example.com Directory Sync filters are shown in Table 2.12.

| Filter type | Filter rule |
|---|---|
| Group filter | `(&(objectCategory=group)(cn=grp-gcp-*))` |
| User filter | `(&(objectCategory=user)(msDS-cloudExtensionAttribute1=googlecloud))` |

**Table 2.12** Google Cloud Directory Sync user and group filters

**Note**: Groups that are synced using Directory Sync need a valid `mail` attribute that has a suffix that matches the Cloud Identity organization name (for example, `group@example.com`). The `mail` attribute does not have to be a valid distribution group for receiving email. But the full email address is what Cloud Identity uses as the unique identifier for the group, and it's how you reference the group when you assign IAM roles.

### 6.2) Users and groups

Groups for example.com follow the naming convention defined earlier in Table 2.2. In the example.com organization, users are not assigned roles directly; instead, groups are the primary method of assigning roles and permissions in Cloud IAM. IAM roles and permissions are assigned during project creation through the deployment pipeline and are then updated as needed.

Users are assigned group membership through the on-premises Active Directory system; as such, group creation and membership is strictly controlled. Table 2.13 shows the groups setup in the example.com foundation. You can set up additional groups as needed on a workload-by-workload basis.

| Group | Description | Roles | Scope |
|---|---|---|---|
| `grp-gcp-billing-viewer@example.com` | Members are authorized to view the spend on projects. Typical members are part of the finance team. | Billing Account Viewer<br>BigQuery Data Viewer<br>BigQuery User | organization for Billing Account Viewer<br>billing project for BigQuery Data Viewer and BigQuery User |
| `grp-gcp-platform-viewer@example.com` | Members have the ability to view resource information across the Google Cloud organization. | Viewer | organization |
| `grp-gcp-security-reviewer@example.com` | Members are part of the security team responsible for reviewing cloud security. | Security Reviewer | organization |
| `grp-gcp-network-viewer@example.com` | Members are part of the networking team and review network configurations. | Compute Network Viewer | organization |
| `grp-gcp-audit-viewer@example.com` | Members are part of an audit team and view audit logs in the logging project. | Logs Viewer<br>Private Logs Viewer<br>Bigquery Data Viewer | logging project |
| `grp-gcp-scc-admin@example.com` | Members can administer Security Command Center. | Security Center Admin Editor | SCC project |

| | | | |
|---|---|---|---|
| `grp-gcp-secrets-admin@example.com` | Members are responsible for putting secrets into Secrets Manager. | Secrets Manager Admin | global secrets project<br>production secrets project<br>non-production project<br>development project |
| `grp-gcp-`*`businesscode-environment-code`*`-developer@example.com` | Groups are created on a per-business code or per-environment basis to manage resources within a particular project. | service- and project-specific permissions | specified project |
| `grp-gcp-platform-operator@example.com` | Members are responsible for platform operations, and they are added into other groups and inherit those permissions. | | |

**Table 2.13** Groups in example.com

There are three types of roles in Cloud IAM:

- Primitive roles include the Owner, Editor, and Viewer roles.
- Predefined roles provide granular access for specific services and are managed by Google Cloud.
- Custom roles provide granular access according to a user-specified list of permissions.

Google recommends using predefined roles as much as possible. IAM recommender should be used to ensure that IAM permissions that are granted to users and groups are not overly permissive. You should eliminate or at least severely limit the use of primitive roles in your Google Cloud organization because the wide scope of permissions inherent in these roles goes against the principles of least privilege. Whenever possible, you should instead use predefined roles, which are designed with inherent separation of duties, and then add custom roles as needed.

**Note**: Groups and users should not have any permissions to alter the foundation components laid out by the deployment pipeline described in Section 5 unless they are one of the privileged identities.

### 6.3) Privileged identities

Privileged identities are accessed through a firecall process, which involves users that are used only in situations that require escalated privileges. Examples of this scenario are when an automated process has broken down, or when permissions need to be elevated temporarily to manually restore a service to a functional state.

| Identity | Description | Role |
|---|---|---|
| `gcp-superadmin@example.com` | Identity that has Google Cloud super administrator permissions. | Super Administrator |
| `gcp-orgadmin@example.com` | Identity that has organization administrator permissions within example.com. | Organization Administrator |
| `gcp-billingcreator@example.com` | Identity that can create billing accounts. | Billing Account Creator |
| `gcp-billingadmin@example.com` | Identity that has billing administrator permissions within example.com. | Billing Account Administrator |
| gcp-*environment*-folderadmin@example.com | Identity (one per folder) that has folder administrator permissions. | Folder Administrator |
| gcp-*businesscode-environment-code*-editor@example.com | Identity (one per business code or per environment) that has project editor permissions for that group's projects in the environment. | Editor |

**Table 2.14** Privileged identities in example.com

Because of the high-level permissions available to privilege identities, access to the privilege identities should require the consent of at least two people within your organization. For example, one person controls access to a privileged identity password, while another person controls access to the associated MFA token.

**Note**: Cloud Identity users with the super admin role, such as `gcp-superadmin@example.com`, bypass the organization's SSO settings and authenticate directly to Cloud Identity. This is to provide access to the Google Admin console in the event of an SSO misconfiguration or outage. For details, see the Super administrator account best practices documentation and the Security best practices for administrator accounts support article.

# 7. Networking

Networking is fundamental to creation of an organization within Google Cloud. This section describes the structure in the example.com reference architecture for VPC networks, projects, IP address space, DNS, firewalls, and connectivity to the example.com on-premises environment.

## 7.1) Shared VPC configuration

Shared VPC is a networking construct that significantly reduces the amount of complexity in network design. A Shared VPC host project can be configured with one or more Shared VPC networks, which can then be leveraged by attached service projects. In this configuration, network policy and control for all networking resources is centralized and easier to manage. Service project departments can configure and manage non-network resources, enabling a clear separation of responsibilities for different teams in the organization.

Resources in Shared VPC networks can communicate with each other securely and efficiently across project boundaries using internal IP addresses. You can manage shared network resources—such as subnets, routes, and firewalls—from a central host project, so you can enforce consistent network policies across the projects.

As shown in Figure 2.7, the example.com architecture uses two Shared VPC networks, base and restricted, as the default networking construct for each environment. Each Shared VPC is contained within a single project. The base VPC network is used for deploying services that contain non-sensitive data, and the restricted VPC network uses VPC Service Controls to limit access to services that contain sensitive data. Access to the restricted VPC network is controlled by Access Context Manager, which has an access policy that grants access based on access levels. By creating an access level, you can also use a base VPC network for hosting services that require access to the sensitive data that's stored in the restricted VPC network.

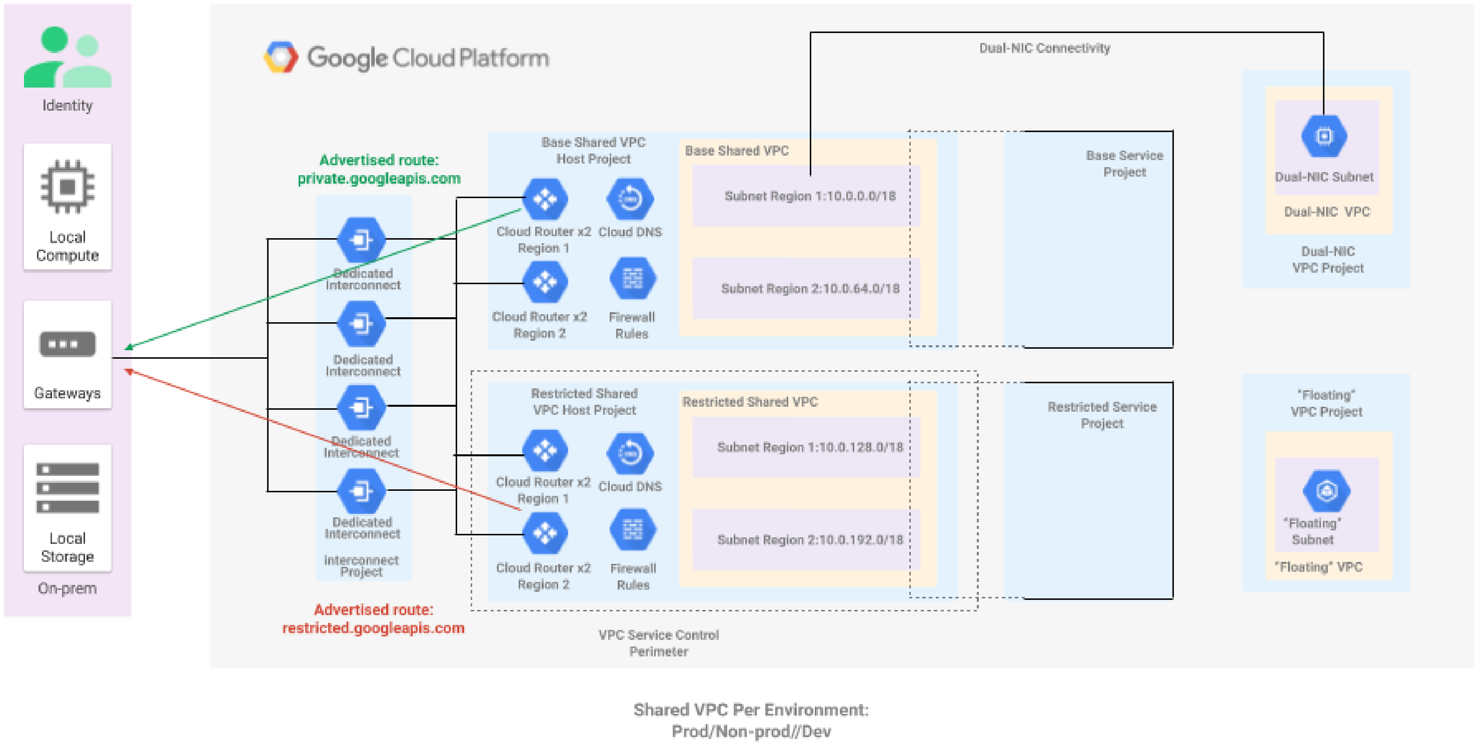


**Figure 2.7** The example.com VPC network structure

Each Shared VPC network has two subnets, with each subnet in a distinct region that spans multiple zones. This infrastructure provides a mechanism for high availability and for disaster-recovery workload configurations. As described in Section 7.2, connectivity with on-premises resources is enabled through four connections for each Shared VPC, using four cloud routers (two per region). Table 2.16 lists the IP address range of each subnet in the example.com design.

**Note:** The example.com network architecture represents one possible architecture for your VPC network design; based on your needs, you might want to consider an alternative architecture for your VPC design.

### 7.1.1) Project deployment patterns

Within the reference example.com organization, there are four patterns for deploying projects: service projects, floating projects, peered VPC projects, and dual-NIC connected projects.

- **Service projects** have been attached to a Shared VPC host project. Service projects are used by example.com for projects that require access to on-premises resources and that need to share resources with other service projects.

- **Floating projects** have no private network connectivity to the example.com on-premises environment, and they require access solely to Google Cloud resources. A modification of the floating projects pattern could be to configure a Shared VPC host project that has no connectivity to on-premises and then to have service projects that have no on-premises connectivity.

- **Peered VPC projects** use VPC Network Peering to enable you to connect VPC networks so that workloads in different VPC networks can communicate internally. Traffic stays within Google's network and doesn't traverse the public internet; this allows you to make services available across VPC networks by using internal IP addresses. Peered VPC networks cannot have a subnet CIDR range in one peered VPC network that overlaps with a static route in another peered VPC network. Only directly peered networks can communicate, because transitive peering is not supported.

- **Dual-NIC connected projects** are connected to either the base VPC network or to the restricted VPC network through a dual-NIC enabled Compute Engine instance. The Compute Engine instance acts as a NAT, allowing resources in the dual-NIC VPC network to communicate to the on-premises environment through the Shared VPC network or with resources within the Shared VPC network. Cross-project multi-NIC configurations require `eth0` in the dual-NIC Compute Engine instance to be connected to the Shared VPC network. Dual-NIC connected projects are used when connectivity either to on-premises or to the Shared VPC network is required, but the IP address space requirements of the dual-NIC enabled VPC network is large.

**Note:** For your workloads that require egress traffic to the internet, the example.com repository contains code that allows you to optionally deploy Cloud NAT instances. For workloads that require ingress traffic from the internet, you should consider using Cloud Load Balancing.

## 7.2) Enterprise-to-Google cloud connectivity

In order to establish connectivity between the on-premises environment and Google Cloud, the example.com reference architecture uses Dedicated Interconnect to maximize security and reliability. A Dedicated Interconnect connection is a direct link between your enterprise's network and Google Cloud. Routing the traffic to Compute Engine instances and to Google Cloud APIs through a private connection reduces the risk of man-in-the-middle attacks that are a threat when traffic is routed through the public internet.

As shown in Figure 2.8, the example.com architecture relies on four Dedicated Interconnect connections in two different metro regions to achieve a 99.99% SLA. The connections are divided into two pairs, with each pair connected to a separate on-premises data center. All connections are hosted in the Interconnect project of the organization. VLAN attachments are used to connect each Dedicated Interconnect instance to Cloud Routers that are attached to the Shared VPC structure described in Section 7.1. Each Shared VPC network has four Cloud Routers, two in each region, with the dynamic routing mode set to global. This enables every Cloud Router to announce all subnets, independent of region. Figure 2.8 also depicts the ASN and IP addresses of the Cloud Routers.

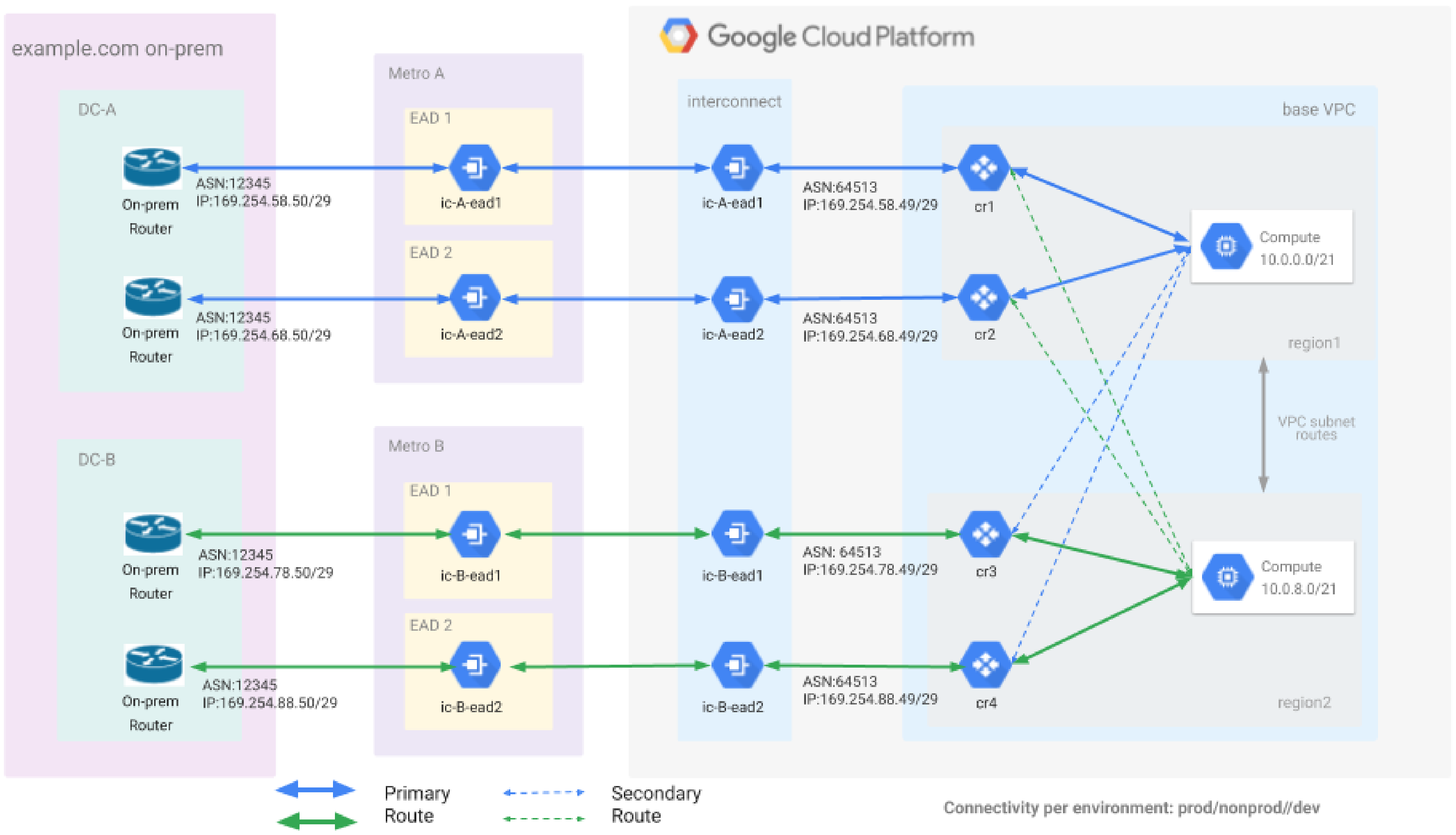


**Figure 2.8** The example.com Dedicated Interconnect connection structure

When you configure filters on the on-premises routers to specify which routes to accept through Border Gateway Protocol (BGP), add the internal IP space that you configured in the previous step. Also accept `199.36.153.8/30` for private Google access and `199.36.153.4/30` for restricted private Google access. For inbound DNS forwarding queries from Google Cloud, you will also need to accept `35.199.192.0/19`, which will be discussed in Section 7.4. Although all of these ranges are public IP addresses, they are not routed over the public internet.

Table 2.15 shows the mapping of Cloud Routers, VLANs, VPCs, connections, regions, and points of presence (POPs) for the production environment.

| Router name | POP-EAD | IC | VLAN attachment | VPC | Region |
|---|---|---|---|---|---|
| cr-p-shared-base-region1-cr1 | metroA-zone1 | ic-a-ead1 | vl-ic-a-ead1-p-shared-base-region1-cr1 | base | region1 |
| cr-p-shared-base-region1-cr2 | metroA-zone2 | ic-a-ead2 | vl-ic-a-ead2-p-shared-base-region1-cr2 | base | region1 |
| cr-p-shared-base-region2-cr3 | metroB-zone1 | ic-b-ead1 | vl-ic-b-ead1-p-shared-base-region2-cr3 | base | region2 |
| cr-p-shared-base-region2-cr4 | metroB-zone2 | ic-b-ead2 | vl-ic-b-ead2-p-shared-base-region2-cr4 | base | region2 |
| cr-p-shared-restricted-region1-cr5 | metroA-zone1 | ic-a-ead1 | vl-ic-a-ead1-p-shared-restricted-region1-cr5 | restricted | region1 |
| cr-p-shared-restricted-region1-cr6 | metroA-zone2 | ic-a-ead2 | vl-ic-a-ead2-p-shared-restricted-region1-cr6 | restricted | region1 |
| cr-p-shared-restricted-region2-cr7 | metroB-zone1 | ic-b-ead1 | vl-ic-b-ead1-p-shared-restricted-region2-cr7 | restricted | region2 |
| cr-p-shared-restricted-region2-cr8 | metroB-zone2 | ic-b-ead2 | vl-ic-b-ead2-p-restricted-region2-cr8 | restricted | region2 |

**Table 2.15** The example.com connection topology

When global routing is enabled and advertised route priority values are left at their default, every Cloud Router announces all subnet routes within its VPC network using BGP. However, routes from remote regions are assigned a higher advertised route-priority value. This means that on-premises routing tables prefer to send traffic over the connection that's closer to the compute region that's being used. In the reverse direction, to keep things symmetrical, Google Cloud penalizes routes that are learned if the routes originate from remote regions. Between two cloud routers in the same region, BGP path selection is left to the configuration of your on-premises routers and BGP sessions.

Traffic flowing in the other direction—from Google Cloud to your on-premises network—also egresses through the connection and through the Cloud Router that's closest to compute resources. Within a single region, there are multiple routes available to on-premises networks that have the same MED value. In those cases, Google Cloud uses ECMP to distribute egress traffic between all possible routes, keeping an affinity based on the hash of a 5-tuple.

**Note:** This section describes using Dedicated Interconnect to connect Google Cloud to the example.com on-premises data centers. Google Cloud also supports Partner Interconnect and Cloud VPN for private connectivity options. You should consider whether those services meet your connectivity requirements.

### 7.3) IP space allocation

In order to properly lay a Google Cloud foundation, you must allocate IP address ranges for the Google Cloud environment. Google Cloud supports a variety of valid IP ranges, including both RFC-1918 and non-RFC-1918 spaces. Note that there are reserved and restricted IP ranges that cannot be used within a subnet. In the example.com reference architecture, IP ranges are assigned to a subnet on project creation.

Table 2.16 provides a breakdown of the IP address space that's allocated for the Shared VPC networks within the example.com reference architecture. The IP address allocation is based on the Google Cloud environment being assigned a continuous RFC 1918 /16 IP address range. Given that there are three environments with the example.com architecture, each environment is allocated a /18 address range that's shared between the base Shared VPC network and the restricted Shared VPC network. Each Shared VPC network assigns a /20 range to allocated networking space and a /20 range to unallocated space. From the allocated IP address space, that range is further broken down into two /21 subnets, with each subnet spanning a region. The unallocated space in the VPC network can be used for peered services (such as Cloud SQL) that require their own IP address space. If RFC 1918 IP address space is limited, non-RFC-1918 address spaces can be used, provided that the on-premises networking can support such topologies.

| VPC | Region | Environment | | |
|---|---|---|---|---|
| | | Production | Non-production | Development |
| Base | Region 1 | `10.0.0.0/21` | `10.0.64.0/21` | `10.0.128.0/21` |
| | Region 2 | `10.0.8.0/21` | `10.0.72.0/21` | `10.0.136.0/21` |
| | Unallocated | `10.0.16.0/20` | `10.0.80.0/20` | `10.0.144.0/20` |
| Restricted | Region 1 | `10.0.32.0/21` | `10.0.96.0/21` | `10.0.160.0/21` |
| | Region 2 | `10.0.40.0/21` | `10.0.104.0/21` | `10.0.168.0/21` |
| | Unallocated | `10.0.48.0/20` | `10.0.112.0/20` | `10.0.176.0/20` |

**Table 2.16** IP address space allocation for Shared VPC

Table 2.17 lists additional IP address ranges that are needed in order to support the example.com reference architecture.

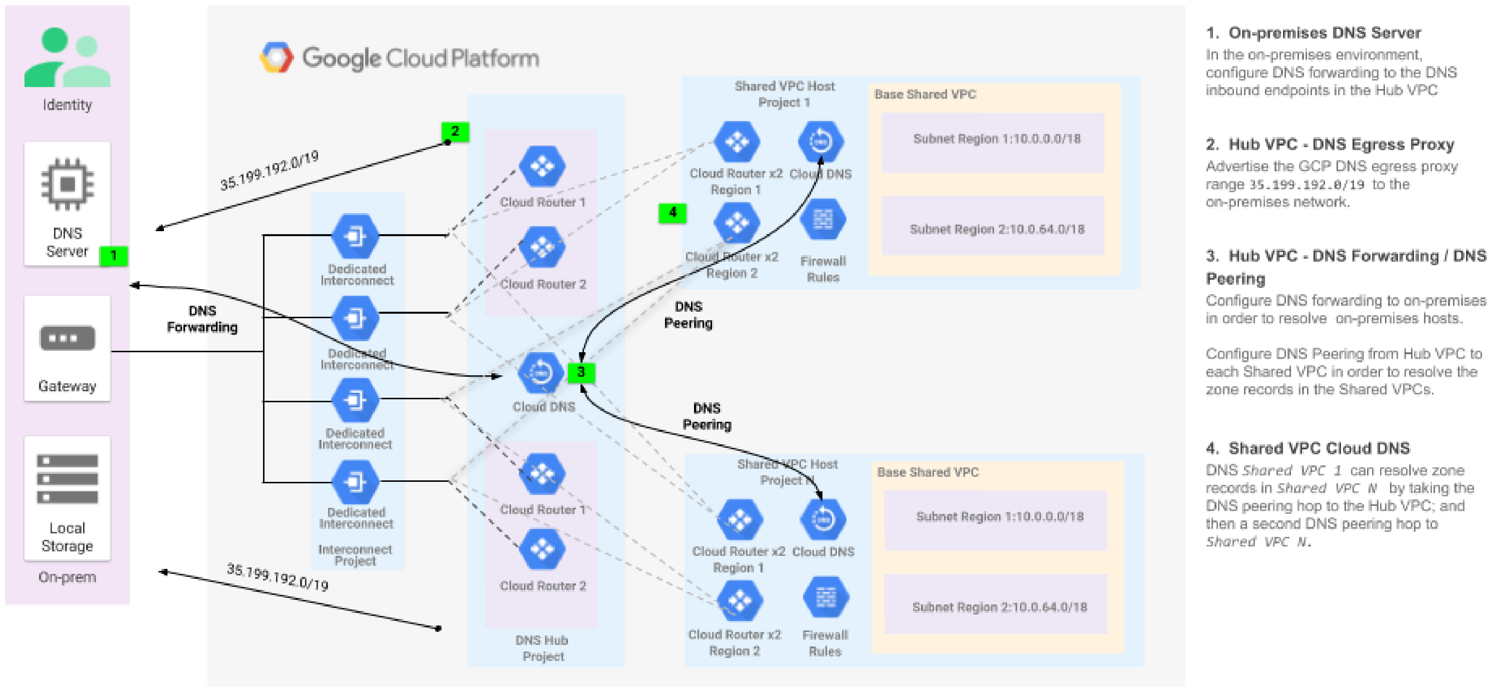
| Project | Description | IP range |
|---|---|---|
| DNS hub | IP space to support independent subnet for Cloud DNS to connect to on-premises. | `172.16.0.0/24` |
| CICD | IP space to support independent subnet for the deployment pipeline. | `172.16.1.0/24` |

**Table 2.17** Additional example.com IP address space allocation

**Note:** Google Cloud supports alias IP ranges and lets you assign ranges of internal IP addresses as aliases to a VM's network interfaces. This is useful if you have multiple services running on a VM and you want to assign each service a different IP address. Alias IP ranges also work with GKE Pods.

### 7.4) DNS setup

Cloud DNS requires `35.199.192.0/19` to be advertised to the on-premises network in order to pass DNS information between Google Cloud and on-premises. To accommodate multiple DNS instances, Cloud DNS has native peering functionality that can be used to create a central hub DNS instance with other DNS instances that are peered to the hub, as shown in Figure 2.9. Cloud DNS peering is a logical construct and does not require network connectivity between projects.



1. **On-premises DNS Server**
   In the on-premises environment, configure DNS forwarding to the DNS inbound endpoints in the Hub VPC

2. **Hub VPC - DNS Egress Proxy**
   Advertise the GCP DNS egress proxy range `35.199.192.0/19` to the on-premises network.

3. **Hub VPC - DNS Forwarding / DNS Peering**
   Configure DNS forwarding to on-premises in order to resolve on-premises hosts.

   Configure DNS Peering from Hub VPC to each Shared VPC in order to resolve the zone records in the Shared VPCs.

4. **Shared VPC Cloud DNS**
   DNS *`Shared VPC 1`* can resolve zone records in *`Shared VPC N`* by taking the DNS peering hop to the Hub VPC; and then a second DNS peering hop to *`Shared VPC N`*.

**Figure 2.9** Cloud DNS setup for example.com

The DNS hub VPC network in the DNS hub project is used as a central VPC network for DNS configuration. The DNS hub project requires connectivity to the on-premises example.com infrastructure, which is enabled through Dedicated Interconnect VLAN attachments and a BGP session that's established from the hub VPC network to on-premises. These VLAN attachments are separate from the VLAN attachments that are used in the Shared VPC networks for the production, non-production, and development environments. The hub VPC network is not attached to the Shared VPC network.

After the DNS hub has been established, the Google Cloud DNS egress proxy range `35.199.192.0/19` needs to be advertised through the Cloud Routers to the on-premises network. In the on-premises environment, the enterprise DNS needs to be configured with DNS forwarding to the DNS inbound endpoints in the DNS hub VPC network.

DNS peering from the DNS hub VPC network to each Shared VPC network is configured so that it resolves the zone records in the Shared VPC networks. Cloud DNS supports transitive DNS peering, but only through a single transitive hop. In other words, no more than three VPC networks (with the network in the middle being the transitive hop) can be involved. For example, consider the architecture in Figure 2.9 and consider two arbitrary VPC networks that need to exchange DNS information: `Shared VPC 1` and `Shared VPC 16`. `Shared VPC 1` can resolve zone records in `Shared VPC 16` by taking the DNS peering hop to the DNS hub VPC and then a second DNS peering hop to `Shared VPC 16`.

## 7.5) On-premises access to Google Cloud APIs through a private IP address using Dedicated Interconnect

In example.com, in order to set up private Google Cloud API access from on-premises hosts that have only private IP addresses, Google Cloud provides two distinct /30 IP endpoints known as virtual IP addresses (VIPs). These VIPs can be accessed over the Dedicated Interconnect connections.

For Google Cloud APIs that are accessible in the base VPC network, on-premises hosts can access those APIs by resolving `private.googleapis.com` to `199.36.153.8/30`. The `private.googleapis.com` VIP enables access to most Google APIs. Although this is a public IP address, it's not announced over the internet by Google. As shown earlier in Figure 2.7, the base VPC network's Cloud Routers announce the `private.googleapis.com` route over the Dedicated Interconnect connection to on-premises hosts.

For Google Cloud APIs that are accessible in the restricted VPC network, on-premises hosts can access those APIs by resolving `restricted.googleapis.com` to `199.36.153.4/30`. The `restricted.googleapis.com` VIP enables access to Google APIs that are accessible in a service perimeter. Although this is a public IP address, it's not announced over the internet by Google. As shown earlier in Figure 2.7, the restricted VPC Cloud Routers announce the `restricted.googleapis.com` route over the interconnect connection to on-premises hosts.

For VMs that run in Google Cloud without external IP addresses, private access to Google Cloud APIs is enabled at project creation through Google Private Access.

**Note:** In order to support both `private.googleapis.com` and `restricted.googleapis.com`, on-premises DNS redirection is needed and must be managed on a workload-by-workload basis, with on-premises workloads directing API calls to either the `private` or `restricted` VIP. In addition, the same DNS indirection and routing needs to be set up in the restricted VPC network for cloud based workloads to take full advantage of VPC Service Controls.

### 7.6) VPC firewall rules

The example.com reference architecture uses VPC firewall rules with network tags in order to restrict traffic to VMs. VPC firewall rules reside in the Shared VPC networks and are deployed through the project deployment pipeline that's described in Section 5. Network tags are added to Google Cloud instances and traffic is evaluated against those tags. By default, example.com uses a deny-all firewall rule with priority 65530 to ensure that all traffic that's not explicitly allowed is denied. For your resources to be able access the `private.googleapis.com` VIP or the `restricted.googleapis.com` VIP you need to create firewall rules which allow traffic to flow to those IP addresses.

Figure 2.10 illustrates how the VPC firewalls with network tags work in the example.com environment.

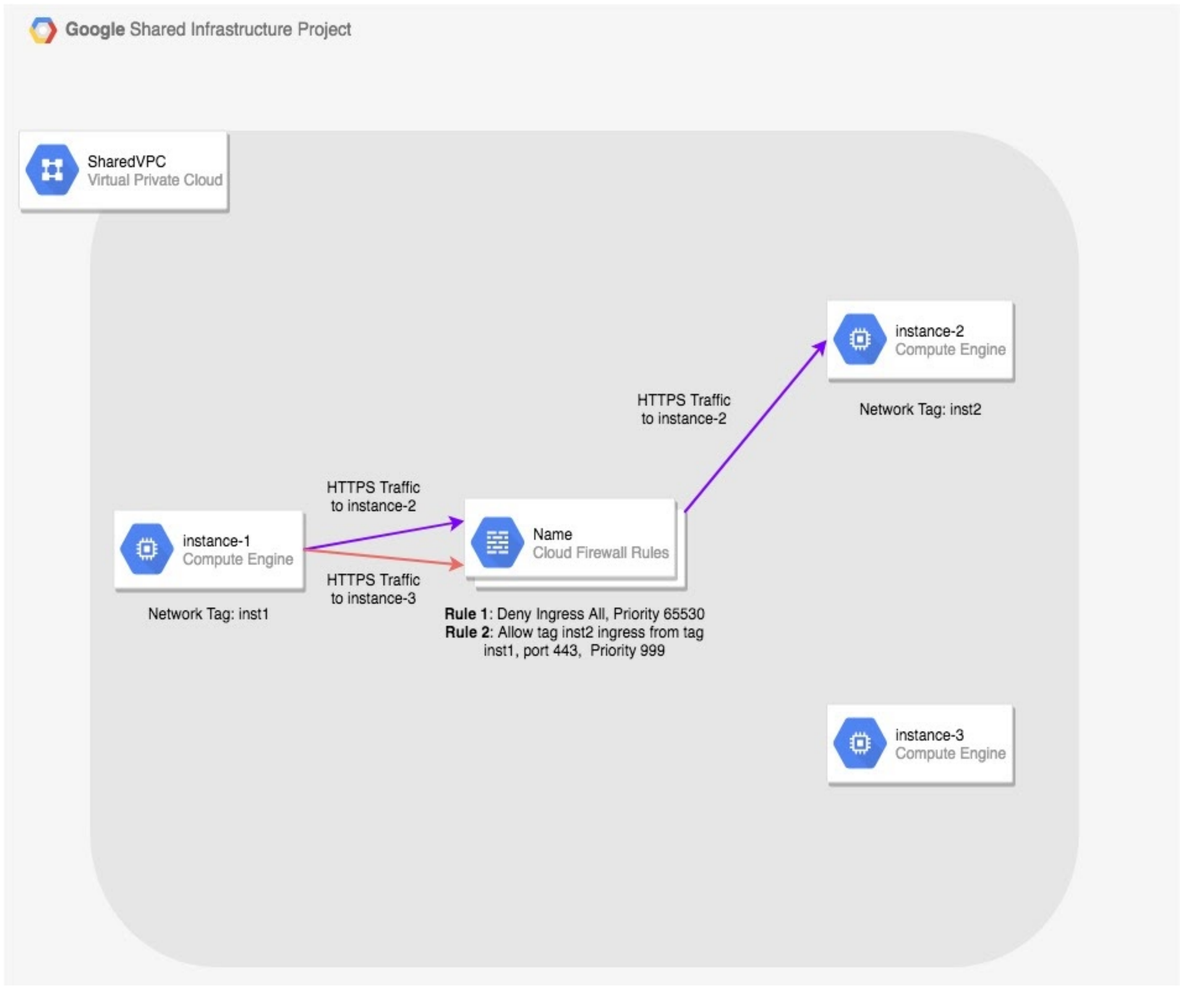

Google Shared Infrastructure Project
SharedVPC
Virtual Private Cloud
instance-2
Compute Engine
Network Tag: inst2
HTTPS Traffic to instance-2
HTTPS Traffic to instance-2
instance-1
Compute Engine
Name
Cloud Firewall Rules
HTTPS Traffic to instance-3
Network Tag: inst1
Rule 1: Deny Ingress All, Priority 65530
Rule 2: Allow tag inst2 ingress from tag inst1, port 443, Priority 999
instance-3
Compute Engine

**Figure 2.10** Firewall rules in example.com

The example.com's default deny-all firewall rule (Rule 1) is applied to all instances. Another firewall rule with a higher priority (Rule 2) allows port 443 traffic ingress to any compute instances that have the tag `https`. That means that the instance named `instance-2` can receive inbound traffic on port 443 because it's tagged with https. In contrast, the instance named `instance-3` does not receive inbound traffic, because it doesn't have a tag that matches a firewall rule.

Network tags can be added or removed from VMs by users with Owner, Editor, or Compute Instance Admin roles. To ensure that network tags are used in a more secure manner, the example.com reference architecture restricts Owner, Editor, and Compute Instance Admin roles from being provided to users or groups except by the firecall process described in Section 6.3.

In cases where the Compute Instance Admin role is required by users or groups, you can create a custom role that has all the permissions of the `compute.instance.admin` role with the exception of the `compute.instances.setTags` permission. Someone with this custom role cannot add or remove tags on VMs; the tags must be added by a controlled mechanism, depending on workload type.

**Note:** You will need to add firewall rules in order to enable different workloads to run within the foundation. For example, if you want to run a Google Kubernetes Engine-based workload with ingress traffic, you will need to enable certain IP ranges to be able to access your Google Kubernetes Engine. Firewall Insights provides visibility into your firewall rule usage and detects firewall rule configuration issues. You can use it to analyze deployed firewall rules, detecting overly permissive firewall rules.

## 8. Secret management

The example.com reference architecture relies on Google Secret Manager to help provide a secure method for storing API keys, passwords, certificates, and other sensitive data. Access to secrets is controlled through Cloud IAM and is managed individually for each organization, folder, project, or secret. Secret Manager has five curated IAM roles that create a default separation of duties:  Secret Manager Secrets Assessor - which enables examination of the payload;  Secret Version Adder - which enables adding new versions to secrets;  Secrets Version Manager - which enables creating and managing versions of secrets; Secrets Manager Viewer - which allows viewing metadata of all Secret Manager resources;  and Secrets Manager Admin - which allows access to administer all Secret Manager resources.

As shown in Figure 2.3, the example.com organization has projects for storing secrets at the organization level and at the environment (folder) level. The intent of having multiple secret repositories is to create a clear separation between organization-level and environment-level secrets. Organization and production secrets are managed by the secrets group described in Section 6.2. It's the responsibility of the secrets group to create secrets, define access policies, and manage the secret's lifecycle.

## 9. Logging

Logging provides important functionality to development organizations, audit organizations, and security organizations, as well as helping to satisfy regulatory compliance. As shown in Figure 2.11, there are a number logging sources in the example.com organization that are aggregated by Google Cloud Logging.

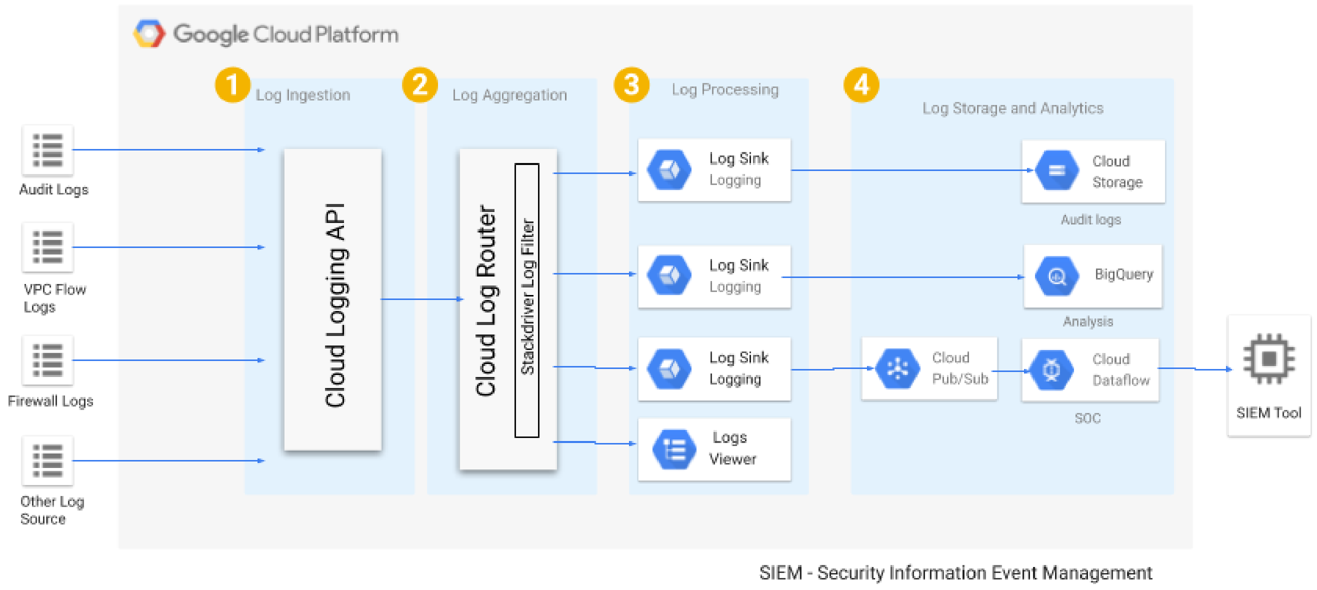


**Figure 2.11** Logging structure for example.com

Within the reference example.com organization, we split logging functions into two projects:

- A standalone logging project named Logging. This project contains Pub/Sub topics, Cloud Functions and a BigQuery instance used for collecting and processing logs generated by the example.com organization.
- A SCC Alert Center project named SCC. This project contains the notification Pub/Sub topics from the Security Command Center. It is separate from the logging project so that you have a clear separation of duties between operations teams that might need general log access and the security team that needs access to specific security information and events.

In Google Cloud, logs are sent to the Cloud Logging API, where they pass through the Logs Router. The Logs Router checks each log entry against existing rules to determine which log entries to discard, which log entries to ingest (store) in Cloud Logging, and which log entries to export through log sinks. Table 2.18 shows the logs that are collected as part of example.com foundation and how those logs are enabled.

| Log source | Description | Management |
|---|---|---|
| Audit Logs | Google Cloud services write audit log entries to these logs to answer the question of "who did what, where, and when?" with Google Cloud resources. | Enabled at an organization level and configured by the pipeline during the initial organization setup. |
| VPC Flow Logs | VPC Flow Logs records a sample of network flows that are sent from and received by VM instances, including instances that are used as GKE nodes. | Enabled for each VPC subnet and configured by the pipeline during project creation. |
| Firewall Rules Logging | Firewall logging creates a record each time a firewall rule allows or denies traffic. | Enabled for each firewall rule and configured by the pipeline during firewall creation. |
| G Suite audit logging | G Suite allows its logs to be shared with the Google Cloud logging service. G Suite collects Login Logs, Admin Logs, and Group Logs. | Enabled at an organization level and configured through the Cloud Identity Admin Console. |
| Data Access audit logs | Data Access audit logs record API calls that read the configuration or metadata of resources, as well as user-driven API calls that create, modify, or read user-provided resource data. | Enabled at an organization level and configured by the pipeline during the initial organization setup. |
| Access Transparency logs | Access Transparency provides logs that capture the actions Google personnel take when accessing your content. | Enabled at an organization level and configured by raising a support case with Google Cloud. |

**Table 2.18** Log sources in example.com

Exporting to a sink involves writing a query that selects the log entries that you want to export and choosing a destination in Cloud Storage, BigQuery, or Pub/Sub. Logging information can be excluded by changing the query. Table 2.19 describes the sinks that are used in the example.com architecture.

| Sink | Description | Purpose |
|---|---|---|
| `sk-c-logging-bq` | Sends the aggregate logs from the organization to a BigQuery table in the logging project. | The aggregated logs can be analyzed in BigQuery and used as part of the detective control architecture that's described in Section 10. |
| `sk-c-logging-bkt` | Sends the aggregate logs from the organization to a Cloud Storage bucket in the logging project. | The aggregated logs can be used for compliance, audit, and incident-tracking purposes. For regulatory purposes, you can apply Cloud Storage Bucket Lock. |

| | | |
|---|---|---|
| `sk-c-logging-pub` | Sends the aggregated logs from the organization to a Pub/Sub topic in the logging project. | The aggregated logs are sent from Pub/Sub to a Dataflow job and from there to an external SIEM, as described in Section 10. |

**Table 2.19** Log sinks in example.com

The sinks and sink destinations are created through the deployment pipeline when the organization objects are first created and configured. When a sink is created, a service account identity is returned; that service account must have permission to write to the specified destination. Those permissions are also configured during setup.

**Note:** In conjunction with logging, you should use SRE concepts (SLI, SLO, SLA) in order to monitor and maintain your environment.

## 10. Detective controls

Detective controls use platform telemetry to detect misconfigurations, vulnerabilities, and potentially malicious activity in the cloud environment. As shown in Figure 2.12, the example.com reference architecture has extensive detective controls to support its overall security posture.

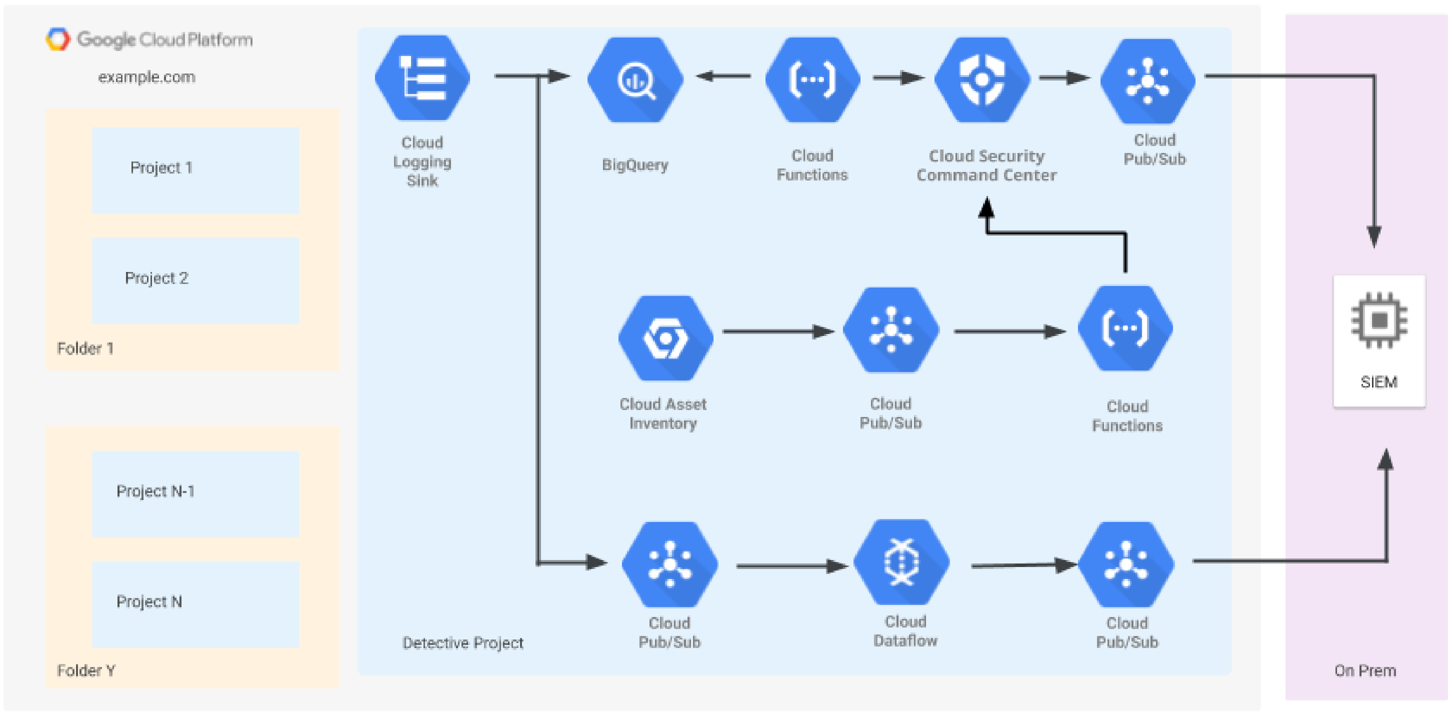


**Figure 2.12** Detective control architecture in example.com

The example.com architecture includes the following detective controls:

- Google security sources through Security Command Center, including security misconfiguration information and vulnerability changes. (For details, see the next section.)
- Connection of Google Cloud logs to Chronicle.
- Asset changes and policy compliance notifications from Cloud Asset Inventory.
- Logs from Google Cloud Logging (including G Suite Audit logs) that are sent to the SIEM tool.
- Custom findings from a serverless BigQuery-based tool.

In addition, if you have existing event sources from any of our third-party providers or have created your own custom sources and custom findings, you can integrate those findings into Security Command Center.

### 10.1) Security Command Center

Security Command Center in Google Cloud provides a single user interface and data platform to aggregate and manage security findings. The example.com architecture uses Security Command Center to aggregate native findings from Google security sources as listed in section 10.1.2 Security sources and custom findings from the BigQuery analysis solution. It then sends Security Command Center notifications to a Pub/Sub topic for the SIEM to consume.

#### 10.1.1) Premium and Standard

Google Security Command Center is available in two versions:

- Security Command Center Premium. This is focused on customers with strong security requirements, especially enterprise customers. It includes a full suite of vulnerability and threat detection capabilities as well as integration options that simplify both the connection to third-party SIEMs and the automation of response and remediation actions. Security Command Center Premium is a paid service.
- Security Command Center Standard. This provides a basic subset of the Security Command Center Premium features for all customers. Its focus is on enabling customers to discover, identify, alert on, manage, and respond to the critical vulnerabilities in their estate. Security Command Center Standard is available to customers free of charge.

Whether you choose Premium or Standard, you can enable the selected Security Command Center capabilities and manage the setup and configuration of the native detectors in the Security Command Center in a unified, centralized way.

The built-in security sources in both Security Command Center Premium and Standard are set by default to apply to all existing and future folders and projects in your Google Cloud organization. Unless there is a specific special security situation, we strongly recommend that you keep this default setting.

#### 10.1.2) Security sources

Security Command Center aggregates findings from a wide variety of security sources, consisting of native Google provided findings, third-party findings, and customer-generated custom findings. Table 2.20 lists the security event sources that are enabled for the example.com reference architecture.

| Security event source | Description |
|---|---|
| Security Health Analytics | Detects common vulnerabilities, misconfigurations, and drift from secure configurations. |
| Anomaly Detection | Uses behavior signals from outside your system to display granular information that helps you detect cloud abuse behavior for your projects and VM instances. |

| | |
|---|---|
| Event Threat Detection | Automatically scans various types of logs for suspicious activity in your Google Cloud environment. |
| Web Security Scanner | Scans public web applications for common application vulnerabilities. |
| Container Threat Detection **(Beta)** | Provides runtime security for GKE containers to help detect reverse shells, suspicious binaries, and libraries. |

**Table 2.20** Security event sources for example.com

**Note:** Security Command Center is also integrated with a number of third-party security sources. In addition, your own applications can generate custom findings in Security Command Center through the findings and sources APIs. If your organization uses third-party security sources or custom findings, you can manage the resulting findings through the Security Command Center dashboard. The findings can also be shared with another system like an enterprise SIEM using Security Command Center API notifications.

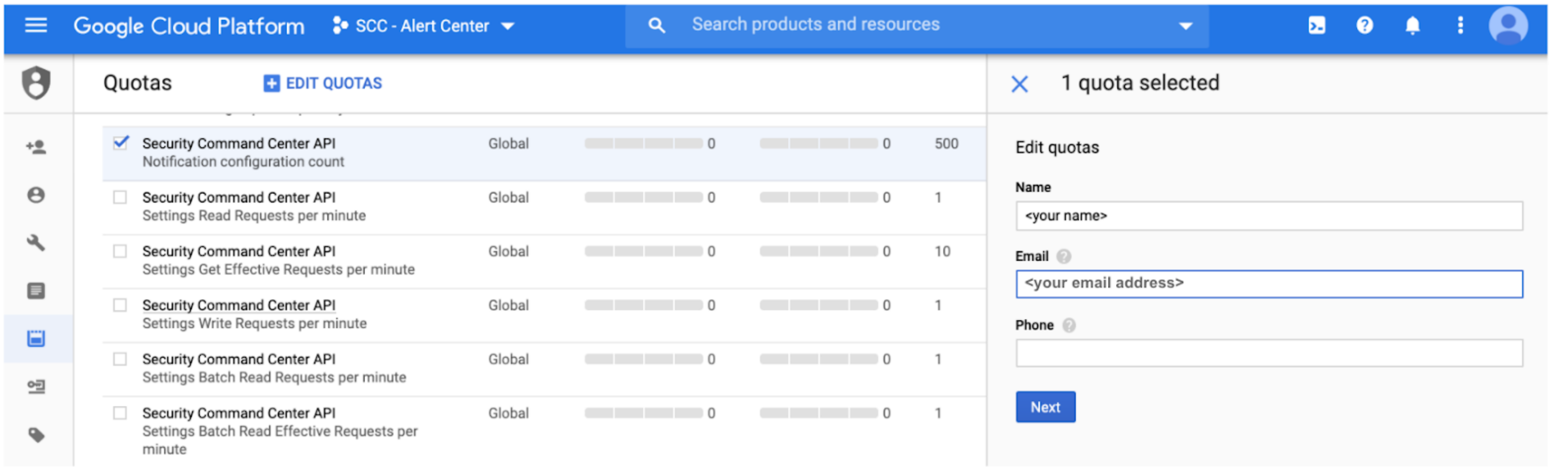
### 10.1.3) Setting up basic security alerting

Beyond detection, it's important for you to be able to take action to respond and remediate detected vulnerabilities and threats. You should leverage the built-in Security Command Center Pub/Sub topic to connect Security Command Center findings to your downstream alerting systems, ticketing, workflow, SOAR systems, and SIEMs. Security Command Center Pub/Sub notifications are managed through a set of notification configs. You can create your notification configs using either the gcloud tool or client libraries and the API. To start, you should create a dedicated SCC Alerts project and restrict IAM access to just the users and services that are allowed to change and manage notification configurations. Within the project, you can then set up the Pub/Sub topics to which you will be publishing events. For each topic, you should further restrict access to that topic to only the necessary individuals and services whose duties require access to the security findings information in the topic.

#### 10.1.3.1) Configuring notifications

After the security-alerting project is created and the topics have been created, you should define your notification configs. For detailed instructions, see Setting up finding notifications in the Security Command Center documentation. For information about managing notification configs, see Creating and managing Notification Configs, and for information about filtering notifications, see Filtering notifications.

The default quota on notification configs is 500 per organization. If you need more, you should submit a request to increase your Security Command Center API quota using the Google Cloud Console, as shown in Figure 2.13.



| | | | | | |
|---|---|---|---|---|---|
| ✓ | Security Command Center API<br>Notification configuration count | Global | 0 | 0 | 500 |
| | Security Command Center API<br>Settings Read Requests per minute | Global | 0 | 0 | 1 |
| | Security Command Center API<br>Settings Get Effective Requests per minute | Global | 0 | 0 | 10 |
| | Security Command Center API<br>Settings Write Requests per minute | Global | 0 | 0 | 1 |
| | Security Command Center API<br>Settings Batch Read Requests per minute | Global | 0 | 0 | 1 |
| | Security Command Center API<br>Settings Batch Read Effective Requests per minute | Global | 0 | 0 | 1 |

Edit quotas

Name: <your name>

Email: <your email address>

Phone:

Next

**Figure 2.13** Request in Cloud Console for additional notification configs quota

The first notification config you should create is one that sends all Security Command Center findings to a Pub/Sub topic that you can then connect to your SIEM. You need the following roles:

- Security Center Admin (broad permissions)
  or
  Security Center Notification Config Editor (specific permissions)
- Pub/Sub Admin on the Pub/Sub topic

After you've set up the roles, follow these instructions:

1. In the Cloud Console, open Cloud Shell, or open a terminal at a computer where you have the Cloud SDK installed and configured.
2. Set up the Pub/Sub topic:

```
# topic-id = siem-export
gcloud pubsub topics create topic-id
export TOPIC_ID=topic-id

# subscription-id = siem-subscription
gcloud pubsub subscriptions create subscription-id --topic topic-id
```

3. Set up the notification config with the filter:

```
export ORGANIZATION_ID=organization-id
export PUBSUB_PROJECT=project-id
export GCLOUD_ACCOUNT=your-username@email.com
```

4. Grant the `gcloud` account the required roles and permissions:

```
gcloud pubsub topics add-iam-policy-binding \
    projects/$PUBSUB_PROJECT/topics/$TOPIC_ID \
    --member="user:$GCLOUD_ACCOUNT" \
    --role='roles/pubsub.admin'

gcloud organizations add-iam-policy-binding $ORGANIZATION_ID \
    --member="user:$GCLOUD_ACCOUNT" \
    --role='roles/securitycenter.notificationConfigEditor'

# The topic to which the notifications are published
PUBSUB_TOPIC="projects/project-id/topics/topic-id"

# The description for the NotificationConfig
DESCRIPTION="Notifies for active findings"

# Filters for all active findings - both vulnerabilities and threats
FILTER="state=\"ACTIVE\""

gcloud alpha scc notifications create notification-name \
    --organization "$ORGANIZATION_ID" \
    --description "$DESCRIPTION" \
    --pubsub-topic $PUBSUB_TOPIC \
    --filter $FILTER
```

For the full set of instructions, including information about how to set up notifications configs using the client libraries and API, see Setting up finding notifications.

#### 10.1.3.2) Matching the notification configurations to your organization's hierarchy

You can use the definition and organization of notification topics to configure the routing of security alerts to the appropriate teams within your organization. Security Command Center enables you to create and store a set of notification configurations, each of which is a unique combination of selection filter and Pub/Sub topic. You match your specific requirements and enforce IAM access policies to security findings by controlling the following:

- The number of notification configs that you create.
- The selection filter in each config.
- The name of the topic where the selected and filtered findings events are published.

The following sections describe four examples of successful notification configuration and topic patterns.

**10.1.3.2.1) One security queue**

The simplest pattern is just one notification config, with the selection filter set for all projects and all findings categories and the target set to just one topic (for example, `gcp-security-alerts`). This pattern is for customers who want all Google cloud security vulnerability and threat findings to be routed to a single centralized Security Operations Center (SOC).

**10.1.3.2.2) By line of business**

You should create multiple separate notification configs when you have adopted the approach where each line of business has its own SOC and therefore owns the end-to-end security responsibilities for your infrastructure, workloads, and applications. Then you should set up a unique Security Command Center notification config and a dedicated topic for each line-of-business SOC. The selection filter in the notification config should select all projects within the scope of the line of business and select all findings categories. This configuration enables IAM control and separation on the dedicated topics between each line-of-business team so that each team has access to and receives only the vulnerability and threat signals relevant to them.

**10.1.3.2.3) Cloud-native DevSecOps**

Similarly, you should create multiple separate notification configs when you have adopted the approach where each application team owns the end-to-end security responsibilities for their infrastructure, workloads, and applications. Then you should set up a unique Security Command Center notification config along with a dedicated topic for each application team. The selection filter in the notification config should select all projects within the scope of the application team and all findings categories. This configuration enables IAM control and separation on the dedicated topics between each application team so each application team has access to and receives only the vulnerability and threat signals relevant to them.

**10.1.3.2.4) By Security finding category**

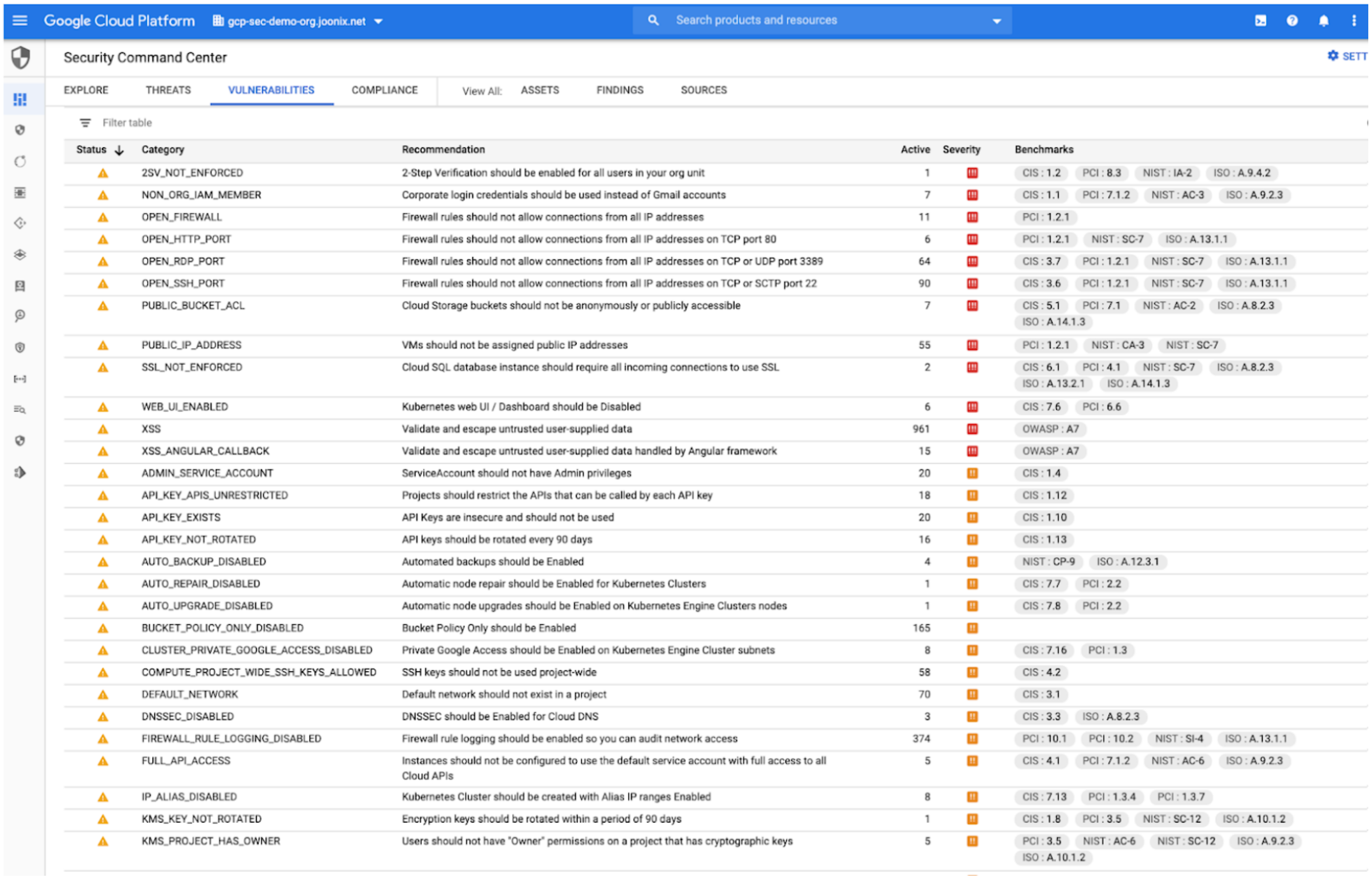
You should create multiple separate notification configs when you have adopted the approach where different types of security findings are handled by different customer teams. The most common example of this is when you have chosen to separate the response and remediation of vulnerabilities and misconfigurations from those of live, active threats. We've seen many customers route the first to their cloud security team and the second to their SOC. In this example case, you can set up a unique Security Command Center notification config and a dedicated Pub/Sub topic for the two teams. The selection filter in the notification config for misconfigurations and vulnerabilities should select all projects and all the finding sources for vulnerabilities (for example, Security Health Analytics and Web Security Scanner). The selection filter in the notification config for threats should select all projects and all the finding sources for threats (for example, Event Threat Detection, Container Threat Detection (**Beta**), and Abuse Anomaly Detection).

### 10.2) Vulnerability and drift detection

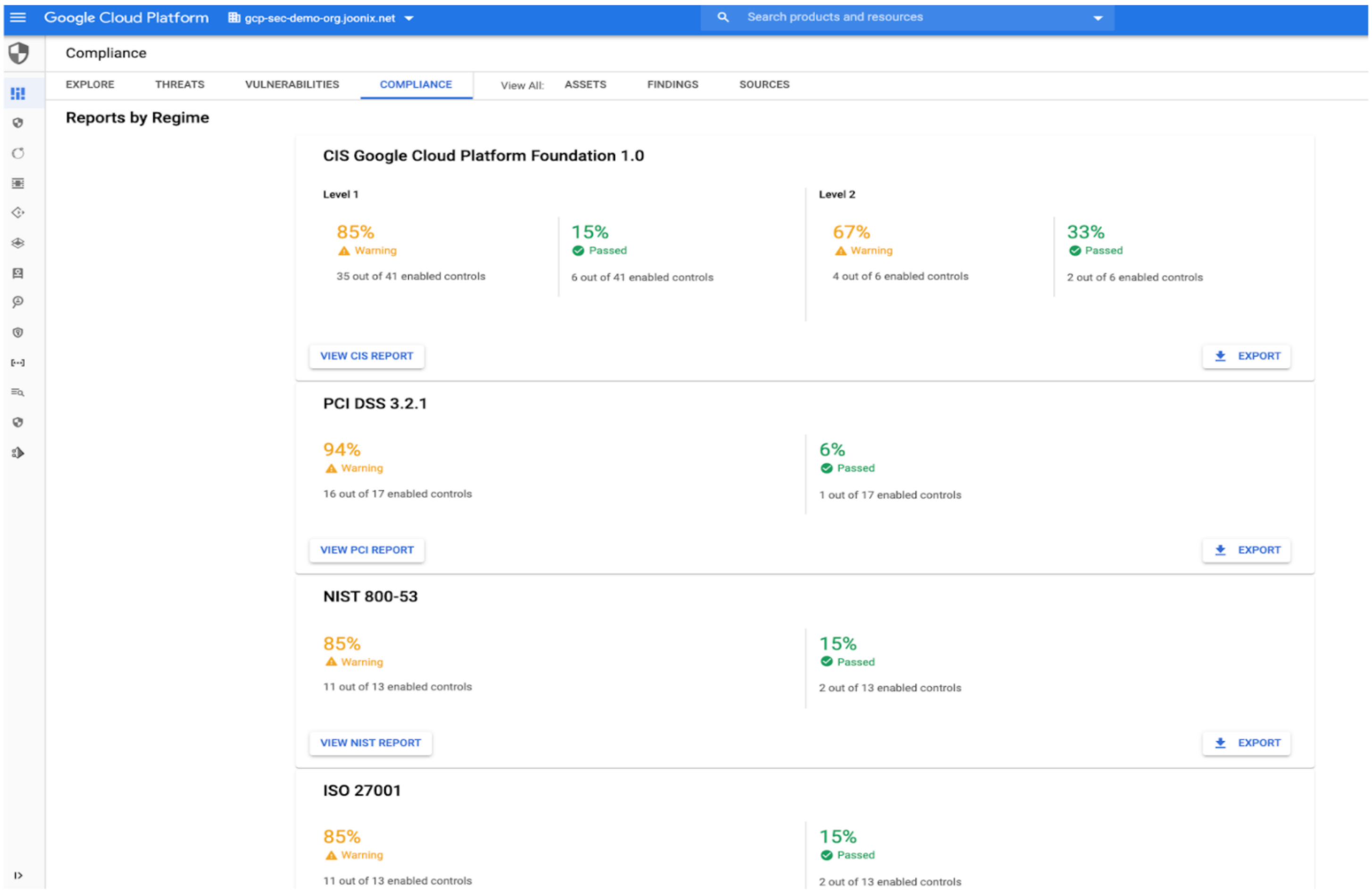
Configuration drift refers to changes in configuration away from a predefined standard or baseline. Over time, as a solution evolves, a certain amount of drift is necessary, but for security or stability reasons, some configuration details should not be violated. Managing drift can be a manual process where an administrator is notified and then takes action where needed. But it can also be automated for specific changes, reverting a change that violates a policy.

#### 10.2.1) Built-in drift detection using Security Command Center Premium

Security Command Center Premium includes Security Health Analytics, which automatically runs over 100 different misconfiguration and vulnerability checks daily against all the Security Command Center-supported resources in your organization. The resulting assessments are listed in the Vulnerabilities tab of the Security Command Center dashboard. You can view and explore the data either at the organization level or for specific sets of projects. You can also further filter the data by status, category, recommendation, severity, active or inactive status, and compliance benchmark. Then in the Compliance tab dashboards, the assessment results are specifically matched against CIS 1.0, NIST-800-53, PCI-DSS, and ISO 27001 benchmarks and summarized for either the full organization or for a specific set of customer-selected projects. Examples of both tabs are shown in Figure 2.14.



| Status | Category | Recommendation | Active | Severity | Benchmarks |
|---|---|---|---|---|---|
| ⚠ | 2SV_NOT_ENFORCED | 2-Step Verification should be enabled for all users in your org unit | 1 | 🟥 | CIS : 1.2 PCI : 8.3 NIST : IA-2 ISO : A.9.4.2 |
| ⚠ | NON_ORG_IAM_MEMBER | Corporate login credentials should be used instead of Gmail accounts | 7 | 🟥 | CIS : 1.1 PCI : 7.1.2 NIST : AC-3 ISO : A.9.2.3 |
| ⚠ | OPEN_FIREWALL | Firewall rules should not allow connections from all IP addresses | 11 | 🟥 | PCI : 1.2.1 |
| ⚠ | OPEN_HTTP_PORT | Firewall rules should not allow connections from all IP addresses on TCP port 80 | 6 | 🟥 | PCI : 1.2.1 NIST : SC-7 ISO : A.13.1.1 |
| ⚠ | OPEN_RDP_PORT | Firewall rules should not allow connections from all IP addresses on TCP or UDP port 3389 | 64 | 🟥 | CIS : 3.7 PCI : 1.2.1 NIST : SC-7 ISO : A.13.1.1 |
| ⚠ | OPEN_SSH_PORT | Firewall rules should not allow connections from all IP addresses on TCP or SCTP port 22 | 90 | 🟥 | CIS : 3.6 PCI : 1.2.1 NIST : SC-7 ISO : A.13.1.1 |
| ⚠ | PUBLIC_BUCKET_ACL | Cloud Storage buckets should not be anonymously or publicly accessible | 7 | 🟥 | CIS : 5.1 PCI : 7.1 NIST : AC-2 ISO : A.8.2.3 ISO : A.14.1.3 |
| ⚠ | PUBLIC_IP_ADDRESS | VMs should not be assigned public IP addresses | 55 | 🟥 | PCI : 1.2.1 NIST : CA-3 NIST : SC-7 |
| ⚠ | SSL_NOT_ENFORCED | Cloud SQL database instance should require all incoming connections to use SSL | 2 | 🟥 | CIS : 6.1 PCI : 4.1 NIST : SC-7 ISO : A.8.2.3 ISO : A.13.2.1 ISO : A.14.1.3 |
| ⚠ | WEB_UI_ENABLED | Kubernetes web UI / Dashboard should be Disabled | 6 | 🟥 | CIS : 7.6 PCI : 6.6 |
| ⚠ | XSS | Validate and escape untrusted user-supplied data | 961 | 🟥 | OWASP : A7 |
| ⚠ | XSS_ANGULAR_CALLBACK | Validate and escape untrusted user-supplied data handled by Angular framework | 15 | 🟥 | OWASP : A7 |
| ⚠ | ADMIN_SERVICE_ACCOUNT | ServiceAccount should not have Admin privileges | 20 | 🟧 | CIS : 1.4 |
| ⚠ | API_KEY_APIS_UNRESTRICTED | Projects should restrict the APIs that can be called by each API key | 18 | 🟧 | CIS : 1.12 |
| ⚠ | API_KEY_EXISTS | API Keys are insecure and should not be used | 20 | 🟧 | CIS : 1.10 |
| ⚠ | API_KEY_NOT_ROTATED | API keys should be rotated every 90 days | 16 | 🟧 | CIS : 1.13 |
| ⚠ | AUTO_BACKUP_DISABLED | Automated backups should be Enabled | 4 | 🟧 | NIST : CP-9 ISO : A.12.3.1 |
| ⚠ | AUTO_REPAIR_DISABLED | Automatic node repair should be Enabled for Kubernetes Clusters | 1 | 🟧 | CIS : 7.7 PCI : 2.2 |
| ⚠ | AUTO_UPGRADE_DISABLED | Automatic node upgrades should be Enabled on Kubernetes Engine Clusters nodes | 1 | 🟧 | CIS : 7.8 PCI : 2.2 |
| ⚠ | BUCKET_POLICY_ONLY_DISABLED | Bucket Policy Only should be Enabled | 165 | 🟧 | |
| ⚠ | CLUSTER_PRIVATE_GOOGLE_ACCESS_DISABLED | Private Google Access should be Enabled on Kubernetes Engine Cluster subnets | 8 | 🟧 | CIS : 7.16 PCI : 1.3 |
| ⚠ | COMPUTE_PROJECT_WIDE_SSH_KEYS_ALLOWED | SSH keys should not be used project-wide | 58 | 🟧 | CIS : 4.2 |
| ⚠ | DEFAULT_NETWORK | Default network should not exist in a project | 70 | 🟧 | CIS : 3.1 |
| ⚠ | DNSSEC_DISABLED | DNSSEC should be Enabled for Cloud DNS | 3 | 🟧 | CIS : 3.3 ISO : A.8.2.3 |
| ⚠ | FIREWALL_RULE_LOGGING_DISABLED | Firewall rule logging should be enabled so you can audit network access | 374 | 🟧 | PCI : 10.1 PCI : 10.2 NIST : SI-4 ISO : A.13.1.1 |
| ⚠ | FULL_API_ACCESS | Instances should not be configured to use the default service account with full access to all Cloud APIs | 5 | 🟧 | CIS : 4.1 PCI : 7.1.2 NIST : AC-6 ISO : A.9.2.3 |
| ⚠ | IP_ALIAS_DISABLED | Kubernetes Cluster should be created with Alias IP ranges Enabled | 8 | 🟧 | CIS : 7.13 PCI : 1.3.4 PCI : 1.3.7 |
| ⚠ | KMS_KEY_NOT_ROTATED | Encryption keys should be rotated within a period of 90 days | 1 | 🟧 | CIS : 1.8 PCI : 3.5 NIST : SC-12 ISO : A.10.1.2 |
| ⚠ | KMS_PROJECT_HAS_OWNER | Users should not have "Owner" permissions on a project that has cryptographic keys | 5 | 🟧 | PCI : 3.5 NIST : AC-6 NIST : SC-12 ISO : A.9.2.3 ISO : A.10.1.2 |



Google Cloud Platform gcp-sec-demo-org.joonix.net

Search products and resources

Compliance

EXPLORE THREATS VULNERABILITIES COMPLIANCE View All: ASSETS FINDINGS SOURCES

Reports by Regime

**CIS Google Cloud Platform Foundation 1.0**

Level 1

85% Warning — 35 out of 41 enabled controls

15% Passed — 6 out of 41 enabled controls

Level 2

67% Warning — 4 out of 6 enabled controls

33% Passed — 2 out of 6 enabled controls

VIEW CIS REPORT EXPORT

**PCI DSS 3.2.1**

94% Warning — 16 out of 17 enabled controls

6% Passed — 1 out of 17 enabled controls

VIEW PCI REPORT EXPORT

**NIST 800-53**

85% Warning — 11 out of 13 enabled controls

15% Passed — 2 out of 13 enabled controls

VIEW NIST REPORT EXPORT

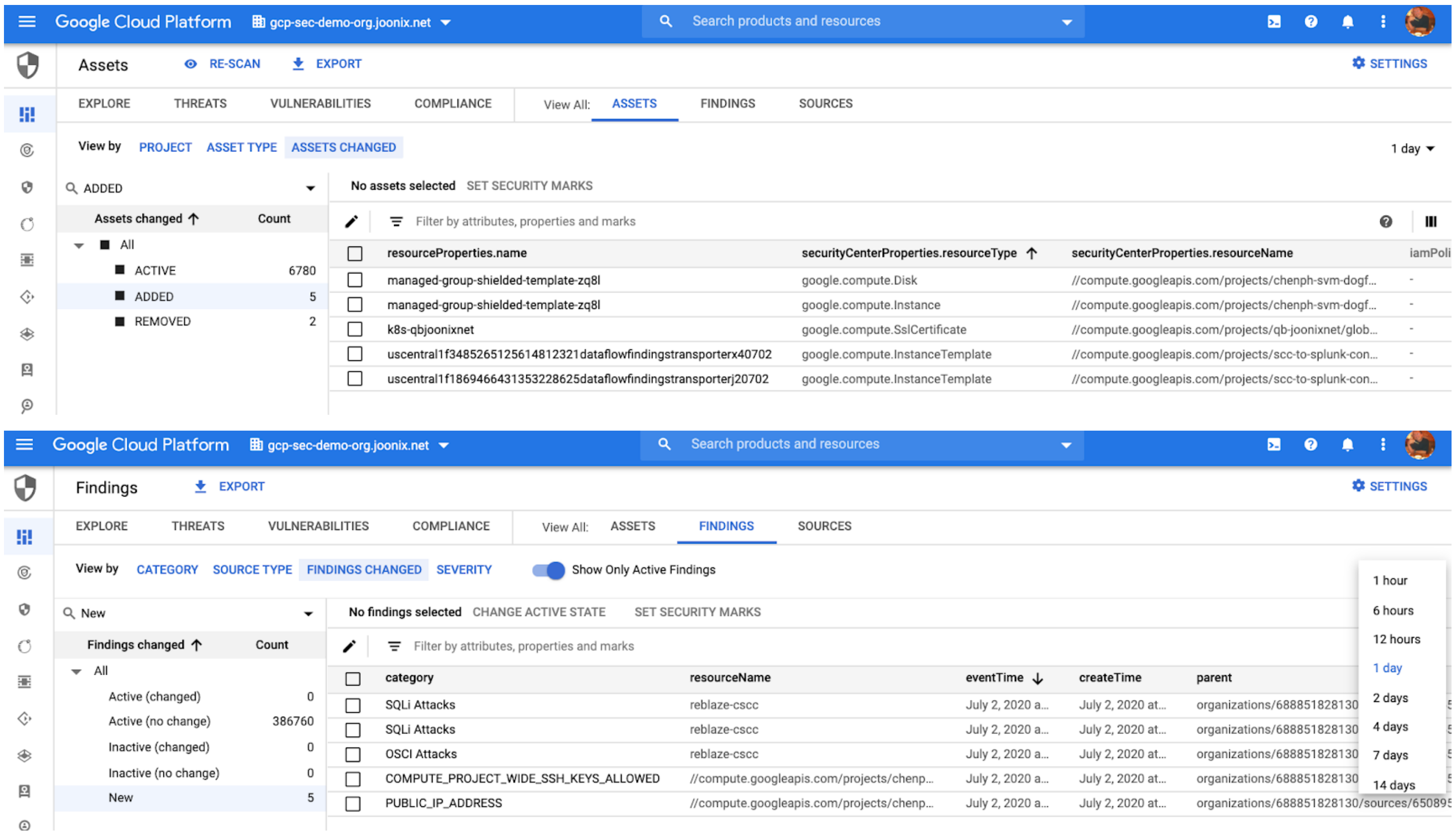
**ISO 27001**

85% Warning — 11 out of 13 enabled controls

15% Passed — 2 out of 13 enabled controls

**Figure 2.14** Vulnerability and Compliance tabs in Security Command Center Premium

With the Security Command Center, you can set up different notification configs to generate alerts on changes in the Security Health Analytics findings to help you identify drift in the configurations of the resources that you care about. In addition, in the Security Command Center dashboard, you can use the Asset Changed view and Findings Changed view to discover and explore changes in user-selected time ranges (for example, last hour, last day, last 7 days, last 30 days, and so on). Examples of both tabs are shown in Figure 2.15.



| Assets changed | Count |
|---|---|
| All | |
| ACTIVE | 6780 |
| ADDED | 5 |
| REMOVED | 2 |

| resourceProperties.name | securityCenterProperties.resourceType | securityCenterProperties.resourceName | iamPoli |
|---|---|---|---|
| managed-group-shielded-template-zq8l | google.compute.Disk | //compute.googleapis.com/projects/chenph-svm-dogf... | - |
| managed-group-shielded-template-zq8l | google.compute.Instance | //compute.googleapis.com/projects/chenph-svm-dogf... | - |
| k8s-qbjoonixnet | google.compute.SslCertificate | //compute.googleapis.com/projects/qb-joonixnet/glob... | - |
| uscentral1f3485265125614812321dataflowfindingstransporterx40702 | google.compute.InstanceTemplate | //compute.googleapis.com/projects/scc-to-splunk-con... | - |
| uscentral1f1869466431353228625dataflowfindingstransporterj20702 | google.compute.InstanceTemplate | //compute.googleapis.com/projects/scc-to-splunk-con... | - |

| Findings changed | Count |
|---|---|
| All | |
| Active (changed) | 0 |
| Active (no change) | 386760 |
| Inactive (changed) | 0 |
| Inactive (no change) | 0 |
| New | 5 |

| category | resourceName | eventTime | createTime | parent |
|---|---|---|---|---|
| SQLi Attacks | reblaze-cscc | July 2, 2020 a... | July 2, 2020 at... | organizations/688851828130 |
| SQLi Attacks | reblaze-cscc | July 2, 2020 a... | July 2, 2020 at... | organizations/688851828130 |
| OSCI Attacks | reblaze-cscc | July 2, 2020 a... | July 2, 2020 at... | organizations/688851828130 |
| COMPUTE_PROJECT_WIDE_SSH_KEYS_ALLOWED | //compute.googleapis.com/projects/chenp... | July 2, 2020 a... | July 2, 2020 at... | organizations/688851828130 |
| PUBLIC_IP_ADDRESS | //compute.googleapis.com/projects/chenp... | July 2, 2020 a... | July 2, 2020 at... | organizations/688851828130/sources/650895 |

**Figure 2.15** The Asset Changed and Findings Changed tabs in the Security Command Center dashboard

In the dashboards, you can explore the time ranges relative to now—that is, to the current time. If instead you want the reference point to be a different time, you can use either the Security Command Center command-line interface or the API or client libraries to set your own reference point.

### 10.2.2) Managed web vulnerability scans

If you deploy web applications, you should also leverage Security Command Center Premium's built-in managed web vulnerability scans. Web Security Scanner currently provides 13 built-in web vulnerability detections that cover 4 of the OWASP top 10. Managed scans automatically run once each week to detect and scan public web endpoints.

### 10.3) Active threat detection

In addition to enabling built-in vulnerability and misconfiguration assessments and threat prevention, you should also leverage Security Command Center Premium's built-in active threat detection capabilities. The initial set that's built in to Security Command Center Premium includes Event Threat Detection and Container Threat Detection (**Beta**).

#### 10.3.1) Event Threat Detection

Event Threat Detection is a built-in service with Security Command Center Premium that detects threats that target your Google Cloud assets. It provides near real-time detection of threats from platform logs, network logs, and compute logs, and it leverages native Google threat intelligence and detection algorithms. Findings are automatically written to the Security Command Center and can also be exported to Cloud Logging. Event Threat Detection performs basic deduplication on findings for users and is enabled by default when you're using Security Command Center Premium.

#### 10.3.2) Container Threat Detection (Beta)

Container Threat Detection is a built-in service with Security Command Center Premium that detects threats that target Google Kubernetes Engine (GKE) containers. It provides near real-time detection of reverse shell execution, suspicious binary execution, and the linking of suspicious libraries. Container Threat Detection is uniquely able to make the connection between containers and underlying physical nodes at the time of threat detection.

To use Container Threat Detection, you must run the latest Container-Optimized OS image for your GKE environment. Container Threat Detection automatically deploys and manages a daemonset container on every node to transmit data plus additional information that identifies the container. Findings are automatically written to Security Command Center and Cloud Logging.

### 10.4) Real-time compliance monitoring of custom policies

To monitor compliance of custom policies in real time, the example.com architecture uses Cloud Asset Inventory real-time notifications. Cloud Asset Inventory can send a Pub/Sub message for each configuration change to a specific asset name or to an asset type. The message triggers a Cloud Function to examine the change. If that change represents a policy violation, the Cloud Function can take action such as reverting the change or notifying an administrator.

One policy that's monitored in example.com is if external Gmail accounts are being granted any permissions to example.com projects. A Cloud Function is used to check when an IAM policy is created or modified that has a Gmail address as a member. If the Cloud Function detects that the policy has been violated, the Cloud Function reverts the change and sends a custom finding to the Security Command Center. Figure 2.16 shows this configuration.

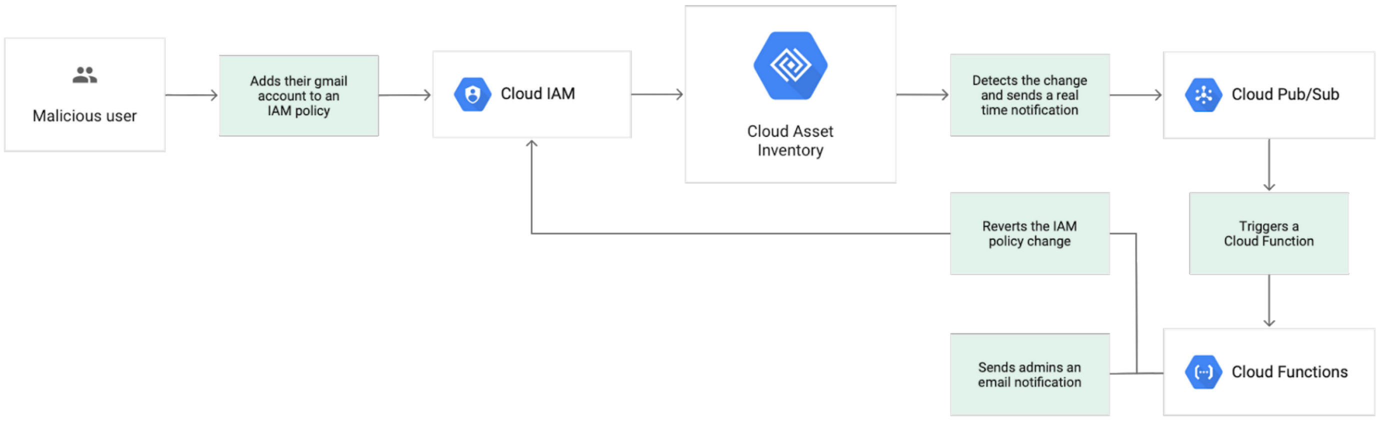

Malicious user
Adds their gmail account to an IAM policy
Cloud IAM
Cloud Asset Inventory
Detects the change and sends a real time notification
Cloud Pub/Sub
Triggers a Cloud Function
Cloud Functions
Sends admins an email notification
Reverts the IAM policy change

**Figure 2.16** Automatically reverting an IAM policy change and sending a notification

Cloud Asset Inventory also enables you to export configuration metadata to a BigQuery table for in-depth analysis. When the data is in BigQuery, you can use SQL queries to explore the configuration and build reports. Listing 2.1 shows an example to detect IAM policies that grant access to a Gmail account.

```
SELECT name, asset_type, bindings.role
FROM `PROJECT_ID.DATASET_ID.TABLE_NAME`
JOIN UNNEST(iam_policy.bindings) AS bindings
JOIN UNNEST(bindings.members) AS members
WHERE members like "%@gmail.com"
```

**Listing 2.1** SQL query to identify IAM bindings with @gmail.com members

### 10.5) Integration with Chronicle

Cloud Logging and Security Command Center events can be exported to Chronicle, which is purpose-built for security threat detection and investigation. Chronicle is built on the Google infrastructure, which lets you take advantage of Google's scale to reduce your time to investigate and triage potential threats. It correlates data from multiple security sources and threat feeds, providing a single timeline-based aggregated view of events and detections. You can leverage Chronicle's detection rules or use real-time search and custom rules to create your own detections.

### 10.6) SIEM solutions integrations

An enterprise SIEM product can be useful for the overall aggregation and visibility of security events, but sometimes the SIEM can be inefficient when dealing with cloud-scale logging.
Both Security Command Center and Cloud Logging can output configurable built-in Pub/Sub event streams. You can therefore choose the option that best fits your needs.

As shown earlier in Figure 2.12, the example.com reference architecture includes a mechanism for a SIEM tool to ingest Google Cloud logs through a Pub/Sub subscription. A log sink in Cloud Logging is used to put all required logs into a raw logs Pub/Sub topic. A Dataflow job subscribes to the raw logs topic and aggregates the logs before putting them into a processed-logs Pub/Sub topic to which the SIEM can subscribe.

**Note:** Different enterprise SIEM tools can integrate with Google Cloud in a number of ways to receive logs, depending on the tool and the log source. Some other integration methods that you can use are the following:

- Direct ingestion from the Security Command Center-native Pub/Sub topic.
- Direct ingestion from a Pub/Sub log sink (if volumes are manageable).
- Flat-file ingestion from Cloud Storage or from a file server.
- Cloud Functions to take Pub/Sub events and turn them into API calls to the SIEM.
- Pulling logs from Google Cloud using Google APIs.

#### 10.6.1) Integrations with Splunk

Both Cloud Logging and Security Command Center events can be directly exported to Splunk and can leverage the Splunk Add-on for GCP for integration. For Cloud Logging, you can set up a logging export to Splunk. For Security Command Center, you can set up a notification config. In both cases, after the Pub/Sub event stream is set up, you can leverage the approved Pub/Sub Splunk Dataflow template to complete the integration.

### 10.7) Analyzing your security data using BigQuery

In cases where exporting to an enterprise SIEM is inefficient, an option is to build your own security analysis solution that's made up of cloud-native tools, including BigQuery, Pub/Sub, Cloud Functions, and Security Command Center. This solution is capable of working with massive log volumes. It can identify security events and forward the events to an enterprise SIEM as findings, rather than as raw logs.

#### 10.7.1) Building your own analysis solution

The example.com reference architecture has some cloud-specific security events that are monitored through the BigQuery-based solution. These security event logs are aggregated and surfaced as custom findings in Security Command Center, and are forwarded to the enterprise SIEM tool through the notifications Pub/Sub topic. This BigQuery-based architecture can handle log sources that have a high log volume, such as data-access logs and VPC Flow Logs. For details, see Figure 2.17.

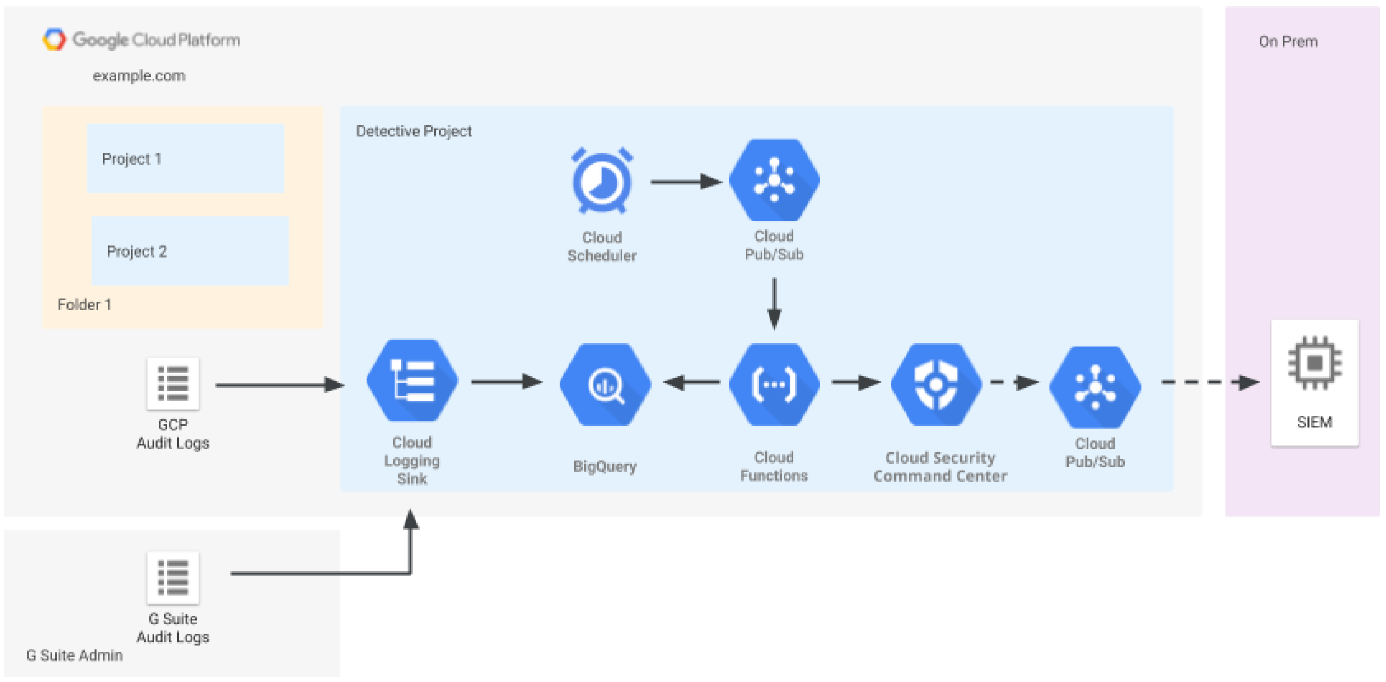


**Figure 2.17** BigQuery-based architecture

The BigQuery-based solution works as follows:

1. Logs from Cloud Logging are stored to a raw-data dataset in BigQuery using a log sink. This dataset can be configured to expire on a regular basis in order to help manage costs.
2. Views are added to a BigQuery dataset that represent the security events being monitored. A view is like a stored query, and in this case, it should produce one entry for each incident of the security event being monitored. Monitored use cases are described in Section 10.7.2.
3. Cloud Scheduler pushes an event to a scheduling Pub/Sub topic every 15 minutes. A Cloud Function is configured to trigger on every scheduling event. The Cloud Function queries the views for new events in the last 20 minutes to ensure no missed events; if it finds events, it pushes them to the Security Command Center API as custom findings.
4. The Security Command Center notifications feature pushes all new findings, including custom and Google or third-party security sources, to a Pub/Sub findings topic.
5. The enterprise SIEM subscribes to this findings topic to receive the security events.

### 10.7.2) Examples of alerting use cases

Alerting use cases are implemented as SQL queries that are saved as views in BigQuery. The following list specifies conditions that generate alerts using the BigQuery-based architecture.

- A login occurs from a high-privilege account login (Super Admin, Organization Admin, and so on).
- Cloud IAM permissions are granted to a user who is from a non-allowed domain.
- Changes are made to logging settings.
- VPC Flow Logs with IP addresses that are outside of expected ranges are detected.
- Changes are made to permissions on critical encryption keys.
- Non-approved Google services are used.

You can extend these use cases by writing a query and saving it as a view in the BigQuery dataset so that it gets queried regularly for new results with views for other use cases. As an example, the following query (Listing 2.2) addresses the case where logging settings are changed.

```
SELECT
  receiveTimestamp,
  timestamp AS eventTimestamp,
  protopayload_auditlog.requestMetadata.callerIp,
  protopayload_auditlog.authenticationInfo.principalEmail,
  protopayload_auditlog.resourceName,
  protopayload_auditlog.methodName
FROM
  `${project}.${dataset}.cloudaudit_googleapis_com_activity_*`
WHERE
  protopayload_auditlog.serviceName = "logging.googleapis.com"
```

**Listing 2.2** SQL query to detect logging setting changes

# 11. Billing

You can use billing as a mechanism to enhance governance and security by monitoring Google Cloud usage and alerting on unexpected consumption. Google Cloud charges are handled by associating projects with billing accounts. The example.com reference architecture uses a single billing account that all projects are associated with when projects are created, as described in Section 5. Table 2.13 lists the groups and roles that are associated with the example.com billing account roles.

### 11.1) Billing alerts

You can set budgets at a project level or organization level, either as a fixed amount to suit steady-state expenditure, or as a percentage of the previous month's spend to suit variable costs. Billing alerts can be applied to different scopes within the organization. For example.com, budgets and associated billing alerts are created when the project is created. Billings alerts are also applied to the example.com billing account to provide organization-level budget alerts.

The example.com reference architecture has billing alerts that are triggered when the project or organization consumption reaches 50%, 75%, 90%, and 95% of the budget threshold. By default, billing alerts are sent to the billing administrator and to billing user email accounts. For example.com, these are a service account and firecall account, so emails to the service account and firecall account need to be forwarded. It's important to note that billing alerts on budget spend do not stop usage when the limit has been met; they are only a notification.

**Note:** If more than the default recipients for billing alerts need to be alerted, you can configure additional recipients through the Cloud Console.

### 11.2) Billing exports and chargeback

The Cloud Console has extensive cost management tools that you can use to view and forecast costs in a variety of formats. In the example.com model, these cost management tools are augmented by exporting all billing records to a BigQuery dataset in the Billing project. The export needs to be enabled through the Cloud Console. The exported billing records include the project label metadata that's assigned to the project during project creation, as listed in Table 2.10. As specified, each project has an associated billing code and contact points that can be used for chargeback. A simple SQL query on the exported dataset, as shown in Listing 2.3, can be used to provide billing information for each business unit within example.com.

```
#standardSQL
SELECT
   (SELECT value from UNNEST(labels) where key = 'business-code') AS bu,
   service.description AS description,
   SUM(cost) AS charges,
   SUM((SELECT SUM(amount) FROM UNNEST(credits))) AS credits
FROM `PROJECT_ID.DATASET_ID.TABLE_NAME`
GROUP BY bu, description
ORDER BY bu ASC, description ASC
```

**Listing 2.3** Example query that groups charges by business unit and service

## 12. General security guidance

The default role automatically assigned for Compute Engine service accounts is Editor, which is very broad. Instead of using these default service accounts, create new service accounts for your projects with tightly scoped permissions that are matched to their use cases. After you've confirmed that the new service accounts work for their respective applications, disable the default Compute Engine service accounts. You can find the default Compute Engine service account in the project-level Cloud IAM section of the Cloud Console. The service account name has the following format:

```
project-number-compute@developer.gserviceaccount.com
```

Similarly, if you use the App Engine service, create a new service account with tightly scoped permissions that are matched to your use case and override the default flow. After you've confirmed that the new service account works for your application, disable the App Engine default service account that was automatically created. You can find the default App Engine service account in the project-level Cloud IAM section of the Cloud Console. The service account name has the following format:

```
your-project-id@appspot.gserviceaccount.com
```

Each Kubernetes Engine node has a Cloud IAM service account associated with it. By default, nodes are given the Compute Engine default service account, which you can find by navigating to the Cloud IAM section of the Cloud Console. As previously discussed, this account has broad access by default, making it useful to a wide variety of applications, but it has more permissions than are required to run your Kubernetes Engine cluster. Instead, strongly consider creating and using a minimally privileged service account to run your Kubernetes Engine cluster. For more information about n hardening your Kubernetes Engine cluster, see the documentation for hardening your cluster's security and the Container Security Blog series.

## 13. Additional updates for the next version

The list of possible security controls you might care about for your specific business scenario can be quite broad. We recognize that this initial version of the security foundations blueprint does not cover everything. Future updates will include topics about additional visibility and controls in network security; data security, encryption options and data classification; cloud-native security application development; and application security.

# III. Summary

This guide provided our opinionated step-by-step guidance for configuring and deploying your Google Cloud estate. We highlighted key decision points and areas of focus, and for each of those we provided both background considerations and discussions of the tradeoffs and motivations for each of the decisions we made. We recognize that these choices might not match every individual company's requirements and business context. You are free to adopt and modify the guidance we've provided.

**We will continue to update this blueprint to stay current with new product capabilities, customer feedback, and the needs of and changes to the security landscape.**

A core Google security principle covered in our BeyondProd paper is to use simple, automated, and standardized change rollout, which emphasizes the use of automation to limit exception cases and to minimize the need for human action. We therefore include a full Terraform repository of the scripts that you can use to automate the majority of the curated steps in this guide. You can run the scripts end-to-end to deploy the full opinionated foundation blueprint. The scripts can also be used and modified individually so that you can leverage parts of the blueprint that are the most relevant for your use case.

Finally, we can provide you with access to a demonstration Google organization using the repository of Terraform automation from start to finish. This gives you a view of what the full security foundations are like after they've been configured and are operational.

**If you would like viewer access to the demonstration organization, please contact your Google Cloud sales team.**

For support questions, reach out to us at security-foundations-blueprint-support@google.com.

For advisory and implementation services Google Cloud is collaborating with Deloitte's industry-leading cyber practice to deliver end-to-end architecture, design, and deployment services to support your cloud security journey.