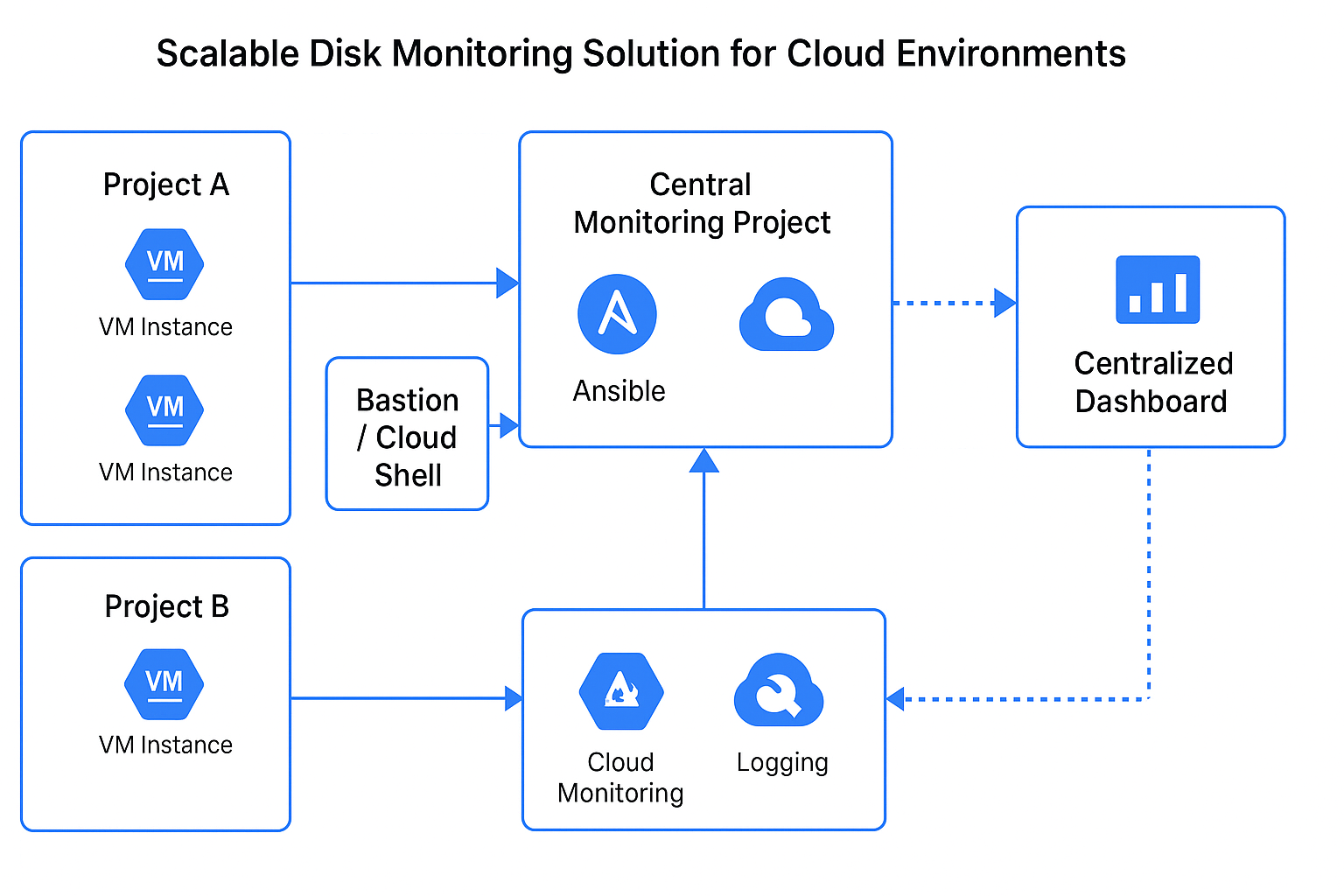
TSA Case Study – *Scalable Disk Monitoring Solution for Cloud Environments*, based on **Google Cloud Platform (GCP)** as the selected provider, and **Ansible** as the configuration management tool.

**✅ 1. High-Level Architecture Diagram**



**✅ 2. Solution Summary**

**🛡 Access Management:**

* **IAM Roles**: Use *least privilege* roles like Compute Viewer, Monitoring Viewer, and Logging Viewer across projects.
* **Service Account**: Create a **centralized service account** with permission to impersonate service accounts in child projects.
* **Shared VPC or VPC Peering**: For connectivity between the monitoring node and VMs across projects.

**🔍 VM Discovery & Enrollment:**

* Use **GCP's Asset Inventory API** or **Cloud Resource Manager API** to dynamically discover VMs across all projects.
* Ansible can be dynamically configured using a **GCP dynamic inventory plugin** (see below).

**✅ 3. Data Collection & Aggregation**

**🛠 Ansible-based Collection:**

Create an Ansible role that:

* SSHs into VMs using service account keys or OS Login
* Runs df -h or lsblk or reads from /proc/mounts and /sys/block
* Sends data to a central Pub/Sub topic, GCS bucket, or directly into Stackdriver/Cloud Monitoring.

**🔧 Sample Ansible Playbook Snippet:**

* + name: Collect disk usage
* hosts: all
* become: yes
* tasks:
  + name: Gather disk usage
* shell: df -h > /tmp/disk\_usage.txt
  + name: Upload to GCS
* gcp\_storage\_object:
* bucket: my-monitoring-bucket
* object\_name: "{{ inventory\_hostname }}-disk-usage.txt"
* src: /tmp/disk\_usage.txt
* credentials\_file: /path/to/credentials.json

**📊 Aggregation:**

* Option 1: Use **Cloud Monitoring Agent** with Ansible to install on VMs → sends disk metrics to **Cloud Monitoring**
* Option 2: Ship results to **Cloud Logging**, and use **Log-based metrics** + dashboards

**✅ 4. Scalability**

**Auto-Discovery with Dynamic Inventory:**

Use Ansible’s GCP inventory plugin (gcp\_compute) for auto-discovery:

**inventory.gcp.yaml**

* plugin: gcp\_compute
* projects:
  + my-project-1
  + my-project-2
* auth\_kind: serviceaccount
* service\_account\_file: /path/to/credentials.json
* filters:
  + status = RUNNING

This ensures **new VMs/projects are automatically included** during execution.

**✅ 6. Summary of Key Components**

| **Component** | **Details** |
| --- | --- |
| **Access Management** | Central service account with delegated access, IAM roles, VPC Peering |
| **Discovery** | Dynamic Inventory via Ansible + GCP API |
| **Data Collection** | Ansible-based agentless SSH collection or cloud-monitoring agents |
| **Aggregation** | Cloud Logging, Monitoring, or GCS/Grafana |
| **Scalability** | Dynamic VM detection, auto-scaling dashboards |
| **Security** | IAM-controlled access, OS Login, encrypted GCS, no hardcoded credentials |