A).

A diagram of a computer

Description automatically generated

Recipe Infrastructure

B). Implement Infrastructure as Code (IaC) to deploy a free-tier file storage service (e.g., AWS S3 bucket or Azure Blob Storage) and a serverless function (e.g., AWS Lambda or Azure Function). The serverless function should be configured to automate the transfer of files from one storage service to another.

Using Terraform, this script would be used to deploy the source storage account, destination storage account on azure with fileshares, and a python script to automate sharein files from the source fileshare to the destination file share.

*# Configure Azure provider*

*provider "azurerm" {*

*features {}*

*}*

*# Resource Groups*

*resource "azurerm\_resource\_group" "rg\_east" {*

*name     = "iot-rg-east"*

*location = "eastus"*

*}*

*resource "azurerm\_resource\_group" "rg\_west" {*

*name     = "iot-rg-west"*

*location = "westus"*

*}*

*# Virtual Networks*

*resource "azurerm\_virtual\_network" "vnet\_east" {*

*name                = "vnet-east"*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*location            = azurerm\_resource\_group.rg\_east.location*

*address\_space       = ["10.0.0.0/16"]*

*}*

*resource "azurerm\_virtual\_network" "vnet\_west" {*

*name                = "vnet-west"*

*resource\_group\_name = azurerm\_resource\_group.rg\_west.name*

*location            = azurerm\_resource\_group.rg\_west.location*

*address\_space       = ["10.1.0.0/16"]*

*}*

*# Subnets*

*resource "azurerm\_subnet" "subnet\_east\_1" {*

*name                 = "subnet-east-1"*

*resource\_group\_name  = azurerm\_resource\_group.rg\_east.name*

*virtual\_network\_name = azurerm\_virtual\_network.vnet\_east.name*

*address\_prefixes     = ["10.0.1.0/24"]*

*}*

*resource "azurerm\_subnet" "subnet\_east\_2" {*

*name                 = "subnet-east-2"*

*resource\_group\_name  = azurerm\_resource\_group.rg\_east.name*

*virtual\_network\_name = azurerm\_virtual\_network.vnet\_east.name*

*address\_prefixes     = ["10.0.2.0/24"]*

*}*

*resource "azurerm\_subnet" "subnet\_east\_3" {*

*name                 = "subnet-east-3"*

*resource\_group\_name  = azurerm\_resource\_group.rg\_east.name*

*virtual\_network\_name = azurerm\_virtual\_network.vnet\_east.name*

*address\_prefixes     = ["10.0.3.0/24"]*

*}*

*resource "azurerm\_subnet" "subnet\_west\_1" {*

*name                 = "subnet-west-1"*

*resource\_group\_name  = azurerm\_resource\_group.rg\_west.name*

*virtual\_network\_name = azurerm\_virtual\_network.vnet\_west.name*

*address\_prefixes     = ["10.1.1.0/24"]*

*}*

*resource "azurerm\_subnet" "subnet\_west\_2" {*

*name                 = "subnet-west-2"*

*resource\_group\_name  = azurerm\_resource\_group.rg\_west.name*

*virtual\_network\_name = azurerm\_virtual\_network.vnet\_west.name*

*address\_prefixes     = ["10.1.2.0/24"]*

*}*

*# ExpressRoute Circuit*

*resource "azurerm\_express\_route\_circuit" "expressroute" {*

*name                  = "expressroute-circuit"*

*resource\_group\_name   = azurerm\_resource\_group.rg\_east.name*

*location              = azurerm\_resource\_group.rg\_east.location*

*service\_provider\_name = "Equinix"*

*peering\_location      = "Silicon Valley"*

*bandwidth\_in\_mbps     = 50*

*sku {*

*tier   = "Standard"*

*family = "MeteredData"*

*}*

*}*

*# IoT Hub*

*resource "azurerm\_iothub" "iothub" {*

*name                = "example-iothub"*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*location            = azurerm\_resource\_group.rg\_east.location*

*sku {*

*name     = "S1"*

*capacity = "1"*

*}*

*}*

*# Event Hub Namespace*

*resource "azurerm\_eventhub\_namespace" "eventhub\_ns" {*

*name                = "example-eventhub-ns"*

*location            = azurerm\_resource\_group.rg\_east.location*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*sku                 = "Standard"*

*capacity            = 1*

*}*

*# Event Hub*

*resource "azurerm\_eventhub" "eventhub" {*

*name                = "example-eventhub"*

*namespace\_name      = azurerm\_eventhub\_namespace.eventhub\_ns.name*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*partition\_count     = 2*

*message\_retention   = 1*

*}*

*# Front Door*

*resource "azurerm\_frontdoor" "frontdoor" {*

*name                = "example-frontdoor"*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*routing\_rule {*

*name               = "routing-rule"*

*accepted\_protocols = ["Http", "Https"]*

*patterns\_to\_match  = ["/\*"]*

*frontend\_endpoints = ["example-frontdoor"]*

*forwarding\_configuration {*

*forwarding\_protocol = "MatchRequest"*

*backend\_pool\_name   = "backend"*

*}*

*}*

*backend\_pool {*

*name = "backend"*

*backend {*

*host\_header = "www.example.com"*

*address     = "www.example.com"*

*http\_port   = 80*

*https\_port  = 443*

*}*

*}*

*frontend\_endpoint {*

*name      = "example-frontdoor"*

*host\_name = "example-frontdoor.azurefd.net"*

*}*

*}*

*# Log Analytics Workspace for Sentinel*

*resource "azurerm\_log\_analytics\_workspace" "workspace" {*

*name                = "example-workspace"*

*location            = azurerm\_resource\_group.rg\_east.location*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*sku                 = "PerGB2018"*

*}*

*# Enable Microsoft Sentinel*

*resource "azurerm\_sentinel\_log\_analytics\_workspace\_onboarding" "sentinel" {*

*workspace\_name      = azurerm\_log\_analytics\_workspace.workspace.name*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*}*

*# Storage Account for Function App and Blob*

*resource "azurerm\_storage\_account" "storage" {*

*name                     = "examplestorage"*

*resource\_group\_name      = azurerm\_resource\_group.rg\_east.name*

*location                 = azurerm\_resource\_group.rg\_east.location*

*account\_tier             = "Standard"*

*account\_replication\_type = "LRS"*

*}*

*# Function App Service Plan*

*resource "azurerm\_service\_plan" "asp" {*

*name                = "example-asp"*

*resource\_group\_name = azurerm\_resource\_group.rg\_east.name*

*location            = azurerm\_resource\_group.rg\_east.location*

*os\_type            = "Windows"*

*sku\_name           = "Y1"*

*}*

*# Function App*

*resource "azurerm\_windows\_function\_app" "function" {*

*name                       = "example-function"*

*resource\_group\_name        = azurerm\_resource\_group.rg\_east.name*

*location                   = azurerm\_resource\_group.rg\_east.location*

*storage\_account\_name       = azurerm\_storage\_account.storage.name*

*storage\_account\_access\_key = azurerm\_storage\_account.storage.primary\_access\_key*

*service\_plan\_id           = azurerm\_service\_plan.asp.id*

*site\_config {}*

*identity {*

*type = "SystemAssigned"*

*}*

*}*

*# Blob Container*

*resource "azurerm\_storage\_container" "container" {*

*name                  = "data"*

*storage\_account\_name  = azurerm\_storage\_account.storage.name*

*container\_access\_type = "private"*

*}*

**C).**

**1. Infrastructure as Code (IaC)**

**Benefits:**

* **Consistency & Reproducibility**: Automated provisioning via IaC reduces manual errors, ensuring all environments are consistent.
* **Automation & Speed**: Infrastructure changes are codified, leading to faster deployments and fewer human errors.
* **Scalability & Flexibility**: Easily scale resources or modify configurations, adapting to business needs without manual intervention.
* **Cost Efficiency**: Optimizes resource allocation by defining infrastructure in code, preventing over-provisioning.

**Pitfalls:**

* **Initial Setup Complexity**: Transitioning from manual processes to IaC requires a learning curve and might be time-consuming.
* **Maintenance Overhead**: If IaC scripts are not properly maintained or refactored, it can lead to technical debt.
* **Security Risks**: Hardcoded credentials or poor security practices in IaC templates can expose infrastructure to vulnerabilities.

**Considerations:**

* **Version Control & Collaboration**: Store IaC scripts in version-controlled repositories like Git for proper change tracking and team collaboration.
* **Security Best Practices**: Use secure secrets management systems (e.g., HashiCorp Vault, AWS Secrets Manager), apply the principle of least privilege, and audit regularly.
* **Monitoring & Testing**: Implement continuous integration/continuous deployment (CI/CD) pipelines that automatically test IaC templates for issues before deployment.

**2. Internet of Things (IoT) & Edge Computing**

**Benefits:**

* **Real-time Decision Making**: IoT sensors collect real-time data, which is processed locally (edge) to make immediate decisions with minimal latency.
* **Scalability**: Edge computing distributes data processing, enabling efficient scaling as the number of IoT devices increases.
* **Bandwidth Optimization**: Data is processed locally, reducing the need to transmit large volumes to centralized data centers or cloud platforms.
* **Reliability**: Critical IoT processes continue functioning even if there’s a network disruption, enhancing system resilience.

**Pitfalls:**

* **Complex Management**: Managing diverse IoT devices and edge nodes at scale involves addressing interoperability, security, and maintenance challenges.
* **Security Vulnerabilities**: IoT devices and edge nodes can be potential attack vectors if not properly secured.
* **Integration with Legacy Systems**: Retro-fitting legacy infrastructure to work with IoT/edge systems may be resource-intensive.

**Considerations:**

* **Data Security & Privacy**: Use encryption and secure communication protocols (e.g., TLS, VPNs) to protect sensitive data in transit.
* **Interoperability**: Adopt open standards (e.g., MQTT, CoAP) to ensure smooth integration across a wide range of IoT devices and platforms.
* **Edge Device Management**: Plan for the lifecycle of edge devices, including remote software updates, patching, and monitoring.

**3. Cost Calculations & Optimizations**

**Benefits:**

* **Cost Transparency**: By understanding cost breakdowns, organizations can more easily pinpoint inefficiencies and adjust resource allocations.
* **Optimized Resource Usage**: Regular cost analysis ensures that organizations only pay for what they use and that resources are scaled appropriately.
* **Predictable Budgets**: By understanding usage patterns, future costs can be better predicted, allowing for more accurate budgeting.

**Pitfalls:**

* **Hidden Costs**: Cloud providers often have complex pricing models with charges for data transfer, storage, and other additional services.
* **Under/Over-provisioning**: Improper resource estimation can lead to unnecessary costs, either due to underutilized resources or inadequate resources that cause performance bottlenecks.
* **Complex Cloud Pricing**: Cloud providers’ intricate pricing models can make it hard to forecast costs accurately, requiring specialized tools and knowledge.

**Considerations:**

* **Cost Management Tools**: Utilize native tools like AWS Cost Explorer, Azure Cost Management, and Google Cloud’s billing tools to track and optimize costs.
* **Monitor Usage & Scaling**: Leverage auto-scaling solutions and regularly audit usage to avoid both resource wastage and under-provisioning.
* **Reserved & Spot Instances**: Maximize cost savings by leveraging reserved instances (for predictable workloads) or spot instances (for flexible, non-critical workloads).

**A Written explanation of how each component of the system will function, including the specific cloud services used.**

I was able to come up with a robust cloud infrastructure to meet the following requirement that enables quick and efficient recipe storage, consolidation, and distribution to our customers, which facilitates creation of Zip-archives of recipes and their exchange in a streamlined and secure way using Azure Cloud.

**Cloud Infrastructure Design:**

A diagram of a computer

Description automatically generated

To meet all the requirement as mentioned in the question:

**1. Storage:**

* **Requirement:** Efficiently handle both large binary files (1+ GB) and small configuration files at scale, with a version control mechanism.
* **Implementation:**
  + **Azure Blob Storage** is used in the architecture for storing large binary files. Blob Storage is designed to store large amounts of unstructured data and can handle files of various sizes.
  + **Azure Files** was used for storing small configuration files. Azure Files provides shared file storage with SMB protocol support, making it easy to manage and access configuration files.
  + Integrated **Azure DevOps** to the architecture for version control. Azure DevOps provides Git-based repositories, enabling tracking and management of changes to these files.

**2. Consolidation:**

* **Requirement:** Consolidate both large and small files into a single archive for easier distribution and securely store the archive in the cloud.
* **Implementation:**
  + **Azure Function** is used to automate the process of consolidating files into a single zip archive. Azure Functions would be triggered based on events (e.g., new file upload) and perform tasks such as zipping files.
  + The consolidated archive would be stored in **Azure Blob Storage** to ensure secure and scalable storage of the zip files.

**3. Distribution:**

* **Requirement:** Enable secure and efficient distribution of the consolidated recipe files to customers, optimized for global reach.
* **Implementation:**
  + **Azure CDN and Front Door (Content Delivery Network)** was added to the architecture to distribute the consolidated recipe files. Azure CDN ensures fast and reliable delivery of content to customers globally by caching content at strategically placed locations and also as a load balancer.
  + Implemented **Azure Role-Based Access Control (RBAC)** to govern access to the recipe files. RBAC allows fine-grained control over who can access specific resources, ensuring secure distribution.
  + **Event Hub** was integrated into the architecture, to handle large volumes of data generated by robotic arms IOT devices when carrying out the recipe, allowing you to ingest, process, and analyze IOT data in real-time.
  + **IOT HUB** was integrated into the architecture to act as a central message hub communication and stand as a Device-to-Cloud and Cloud to Device Communication with the IOT device, in this case the ROBOT arm carrying out the recipe.

**4. Network:**

* **Requirement:** Establish a secure connection between the on-premises network and the cloud infrastructure via VPN.
* **Implementation:**
  + Set up an **Azure VPN Gateway** to create a secure and reliable connection between the on-premises network and the Azure cloud infrastructure. The VPN Gateway supports site-to-site, point-to-site, and VNet-to-VNet connections.
  + **Azure ExpressRoute** To enhance the security and reliability of the network connection between the on-premises environment and the cloud infrastructure, I leveraged Azure ExpressRoute. ExpressRoute offers a private connection to Microsoft Azure and other Microsoft cloud services, which is more reliable and faster compared to standard VPN connections over the internet.

**5. Security:**

* **Requirement:** Address security considerations with data encryption during storage and transmission.
* **Implementation:**
  + In other to secure the zip files at rest, **Azure Key Vault** was integrated into the architecture for managing cryptographic keys and secrets. Also used to ensure secure key management and supports encryption of data at rest.
  + Enable **Azure Storage Service Encryption (SSE)** for data stored in Blob Storage and Files. SSE provides automatic encryption of data at rest using Microsoft-managed keys.
  + Use **Transport Layer Security (TLS)** to encrypt data during transmission, ensuring secure communication between clients and the cloud infrastructure.
  + **Azure Sentinel** was integrated into the architecture because itprovides a complete SIEM solution that enables proactive threat detection, investigation, and response. By centralizing and analyzing security data, it helps in quickly identifying and responding to potential threats.

**Conclusion:** The cloud infrastructure design leverages various Azure services to meet the requirements of storing, consolidating, and distributing automation recipes securely and efficiently. By using a combination of Azure Blob Storage, Azure Files, Express-route, Azure DevOps, Azure Functions, Azure CDN, Azure VPN Gateway, Azure Key Vault, Sentinel and encryption mechanisms, the proposed architecture ensures a robust and scalable solution for managing automation recipes.

Based on the information above, do this A written explanation of how each component of the system will function, including the specific cloud services and technologies you would use.