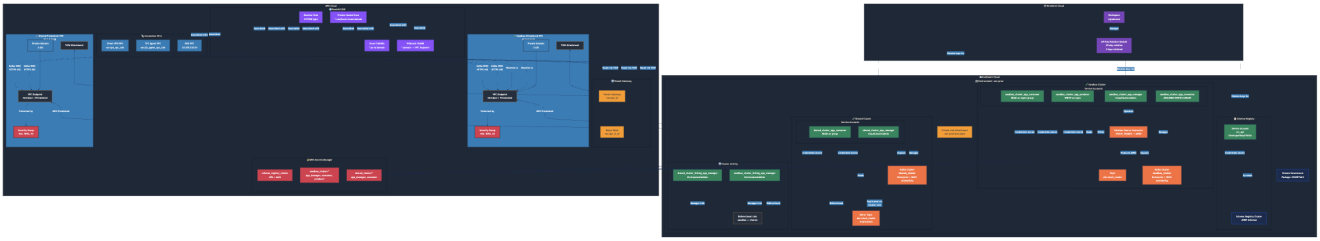
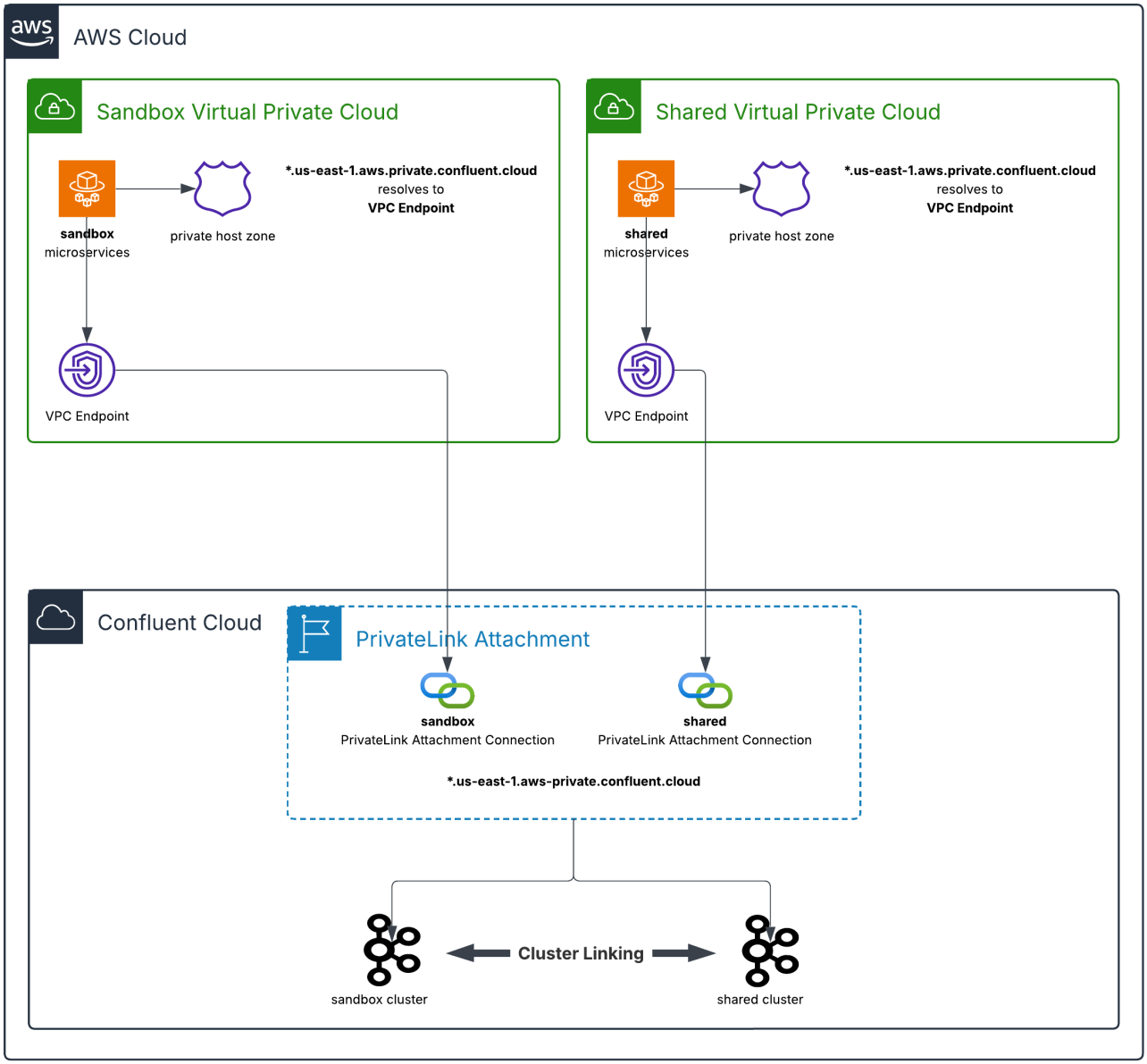
IaC Confluent Cloud AWS Private Linking with Cluster Linking Example



This repository provides **production-grade Terraform infrastructure-as-code** that implements a **secure, multi-network Confluent Cloud architecture**. It demonstrates **AWS PrivateLink connectivity from a single Confluent Cloud environment to multiple AWS VPCs**, enabling private, network-isolated access without exposing traffic to the public internet.

The solution also showcases **in-region Cluster Linking between two Confluent Cloud Kafka clusters**, enabling **low-latency, fully managed data replication** across teams, lines of business, or isolated environments (for example, development, staging, and production) within the same AWS region.

Cluster Linking maintains an **in-sync mirror of selected topics** on the consuming cluster. This isolation allows consuming teams to independently scale **large numbers of consumers, stream processing applications, and downstream sinks** without impacting the producing cluster. From the producer's perspective, the load is equivalent to **a single additional consumer**, regardless of downstream scale.



Access control and ownership remain cleanly separated: the producing team grants **scoped read credentials** to approved topics, while the consuming team **creates, owns, monitors, and manages the cluster link**. This pattern enables secure, scalable data sharing with clear operational boundaries and minimal coupling.

Below is the Terraform resource visualization of the infrastructure that's created:

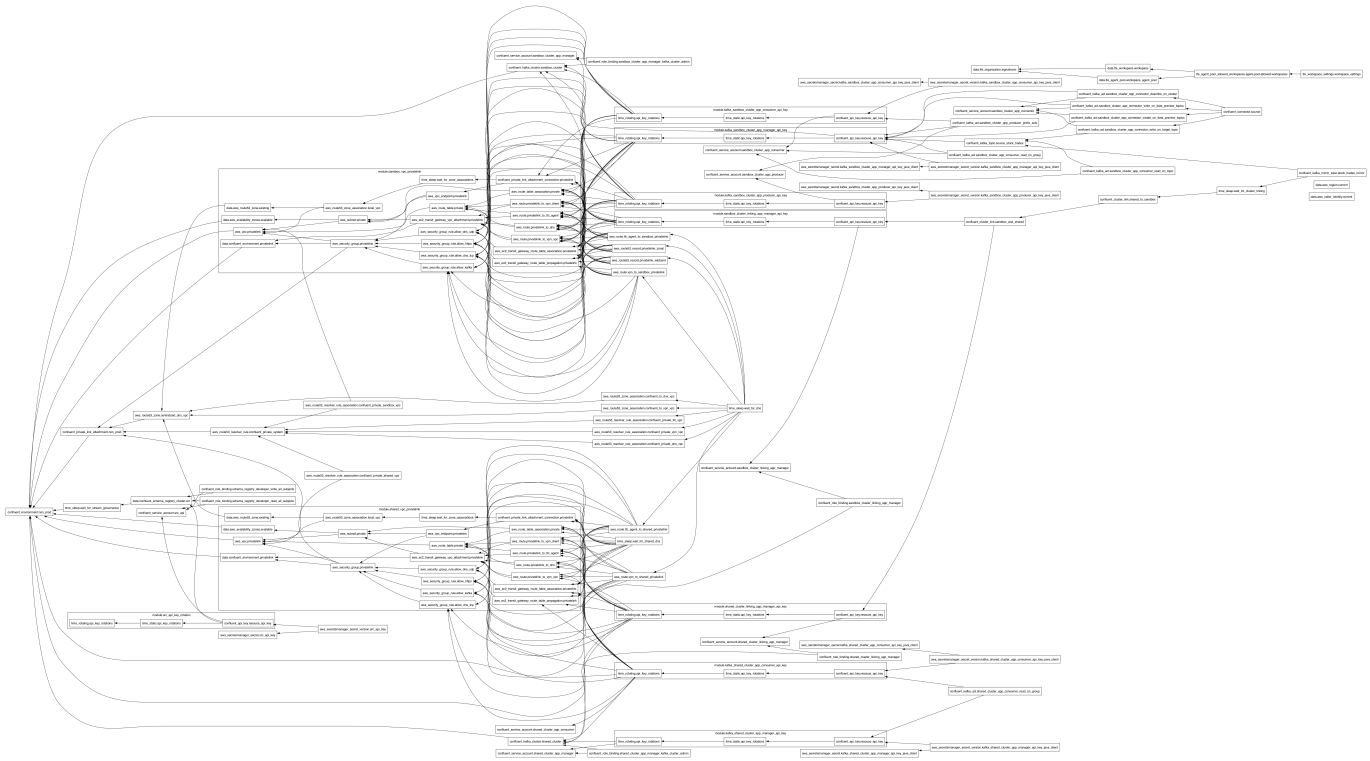


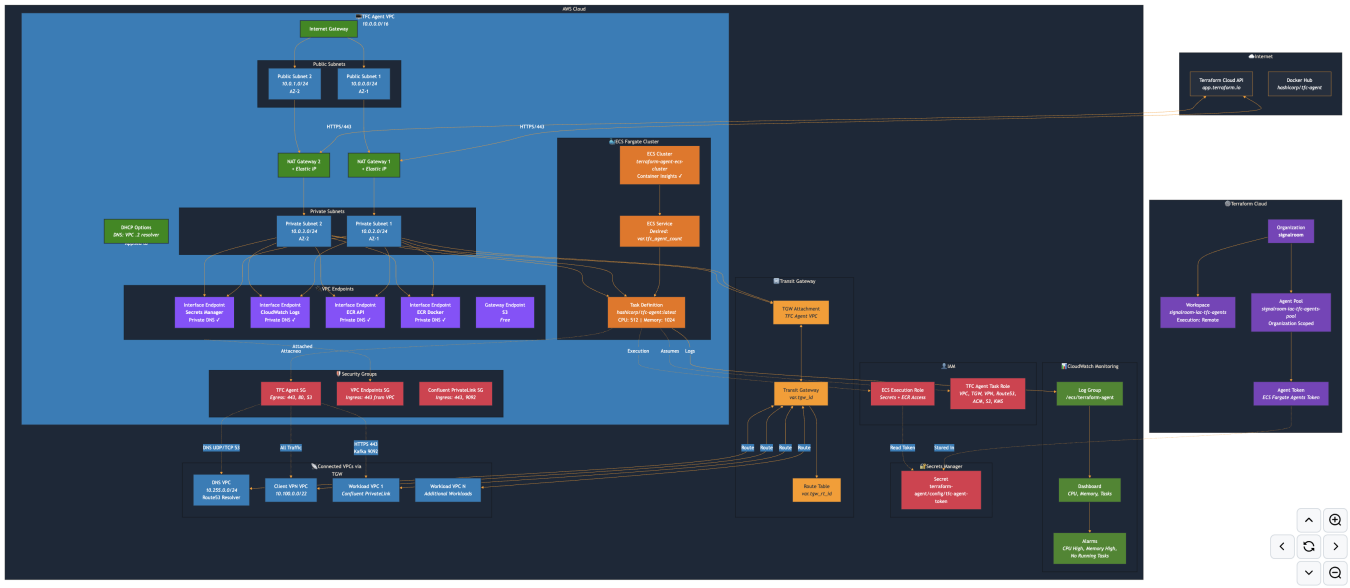
Table of Contents

1.0 Prerequisites

This project assumes you have the following prerequisites in place:

- Client VPN, Centralized DNS Server, and Transit Gateway
- Terraform Cloud Agent

1.1 Client VPN, Centralized DNS Server, and Transit Gateway



The diagram illustrates:

Core Components:

- **Transit Gateway Hub** — Central connectivity point with ASN 64512, DNS support, and VPN ECMP enabled. Includes main and additional route tables for traffic isolation
- **Client VPN VPC** — Hosts the VPN endpoint with mutual TLS authentication using ACM server/client certificates, with TGW attachment for routing

Connected VPCs:

- **Workload VPCs** — Multiple VPCs with distinct CIDR ranges connected via TGW attachments
- **TFC Agent VPC** — Dedicated VPC for Terraform Cloud agents (local execution mode)
- **DNS VPC** — Centralized Route 53 resolver for cross-VPC DNS resolution

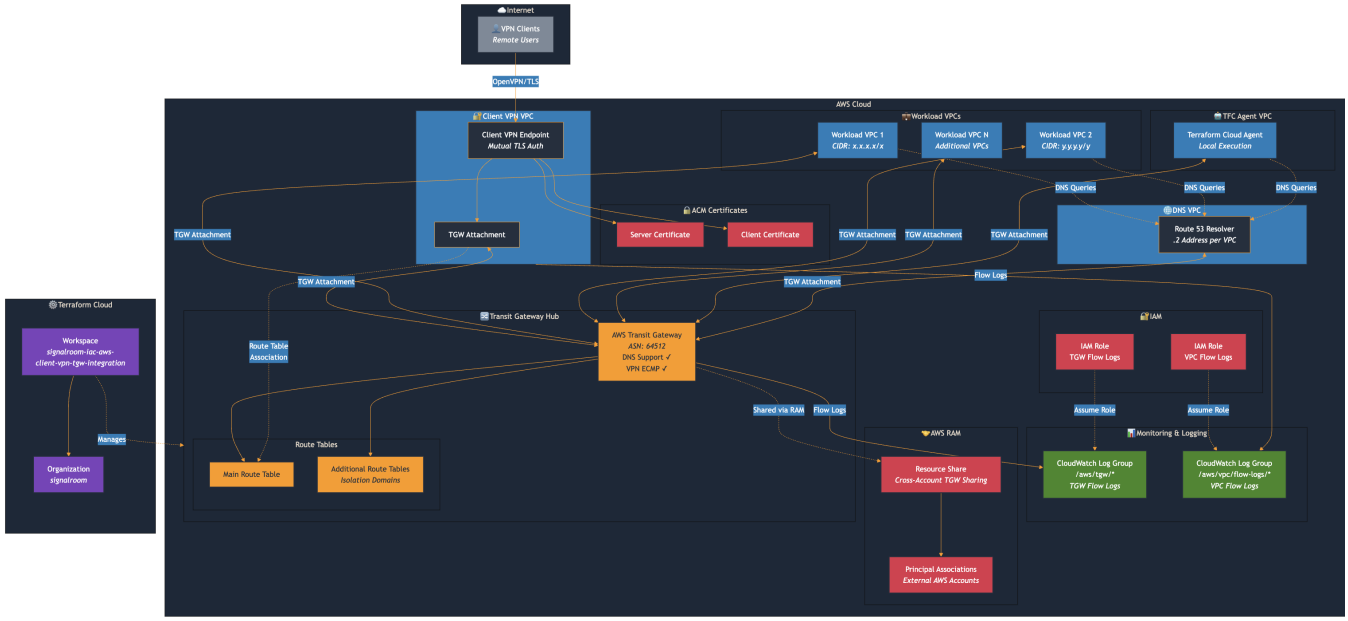
Supporting Services:

- **AWS RAM** — Cross-account sharing of the Transit Gateway to external AWS accounts
- **CloudWatch** — Flow logs for both TGW and VPC traffic with dedicated IAM roles
- **Terraform Cloud** — signalRoom organization managing the infrastructure via the `signalroom-iac-aws-client-vpn-tgw-integration` workspace

Traffic Flow:

1. Remote VPN clients connect via OpenVPN/TLS to the Client VPN endpoint
2. Traffic routes through the TGW to reach workload VPCs, TFC agents, or DNS services
3. VPN client traffic routes back through the TGW via the dedicated route table association

1.2 Terraform Cloud Agent



The diagram illustrates:

Terraform Cloud Integration:

- **signalroom** organization with remote execution workspace
- **Agent Pool** (organization-scoped) with token stored in AWS Secrets Manager
- Agents poll Terraform Cloud API via NAT Gateway

TFC Agent VPC (10.0.0.0/16):

- **Public Subnets** — Host NAT Gateways with Elastic IPs for outbound internet
- **Private Subnets** — Run ECS Fargate tasks (no public IPs)
- **DHCP Options** — Configure DNS to use VPC's .2 resolver

ECS Fargate Cluster:

- Runs `hashicorp/tfc-agent:latest` containers (512 CPU / 1024 MB)
- Container Insights enabled for monitoring
- Circuit breaker with auto-rollback on deployment failures

VPC Endpoints (PrivateLink):

- **Secrets Manager** — Retrieve TFC agent token privately
- **CloudWatch Logs** — Stream logs without NAT
- **ECR API + Docker** — Pull container images privately
- **S3 Gateway** — Free endpoint for ECR layer storage

Transit Gateway Connectivity:

- Routes to **DNS VPC** (10.255.0.0/24) for Confluent domain resolution
- Routes to **Client VPN VPC** (10.100.0.0/22) for admin access
- Routes to **Workload VPCs** for Confluent PrivateLink (ports 443, 9092)

Security Groups:

- **TFC Agent SG** — Outbound HTTPS/HTTP/DNS to internet + specific VPC routes
- **VPC Endpoints SG** — Inbound 443 from VPC CIDR
- **Confluent PrivateLink SG** — Inbound 443/9092 from agents

2.0 Project's Architecture Overview

Confluent Cloud Environment (non-prod):

Component	Details
Sandbox Cluster	Enterprise tier, HIGH availability, hosts `dev-stock_trades` topic
Shared Cluster	Enterprise tier, HIGH availability, receives mirrored data
Cluster Linking	Bidirectional link replicates `dev-stock_trades` between clusters
DataGen Connector	Produces STOCK_TRADES data in AVRO format
Schema Registry	Stream Governance ESSENTIALS package for schema management
PrivateLink Attachment	Single attachment exposes both clusters to AWS

Service Accounts & RBAC:

- **Cluster Managers** — CloudClusterAdmin role for each cluster
- **Producers/Consumers** — Topic-specific ACLs (READ/WRITE/DESCRIBE)
- **Connector SA** — DESCRIBE cluster, WRITE/CREATE topics
- **Cluster Linking SAs** — EnvironmentAdmin for link management
- **Schema Registry SA** — DeveloperRead/Write on all subjects

AWS PrivateLink VPCs:

- **Sandbox VPC** (10.0.0.0/20) — 3 AZ private subnets with VPC Endpoint
- **Shared VPC** (10.1.0.0/20) — 3 AZ private subnets with VPC Endpoint
- Both attached to Transit Gateway with route propagation

DNS Architecture:

- **Private Hosted Zone** — Centralized PHZ for Confluent domain
- **Wildcard + Zonal CNAMEs** — Route to VPC Endpoint DNS entries
- **SYSTEM Resolver Rule** — Associated with all 5 VPCs (DNS, VPN, TFC Agent, Sandbox, Shared)

Security & Secrets:

- **Security Groups** — Allow ports 443 (HTTPS), 9092 (Kafka), 53 (DNS) from TFC Agent and VPN CIDRs
- **Secrets Manager** — Stores JAAS configs and bootstrap servers for all service accounts
- **API Key Rotation** — 30-day rotation with 2 keys retained per service account

Connectivity Flow:

1. VPN/TFC Agent clients resolve `*.<AWS_REGION>.aws.private.confluent.cloud` via PHZ
2. DNS returns VPC Endpoint private IPs

3. Traffic routes through Transit Gateway to appropriate PrivateLink VPC
4. VPC Endpoint forwards to Confluent Cloud via AWS PrivateLink

3.0 Resources

3.1 Terminology

- **PHZ**: Private Hosted Zone - AWS Route 53 Private Hosted Zone is a DNS service that allows you to create and manage private DNS zones within your VPCs.
- **TFC**: Terraform Cloud - A service that provides infrastructure automation using Terraform.
- **VPC**: Virtual Private Cloud - A virtual network dedicated to your AWS account.
- **AWS**: Amazon Web Services - A comprehensive cloud computing platform provided by Amazon.
- **CC**: Confluent Cloud - A fully managed event streaming platform based on Apache Kafka.
- **PL**: PrivateLink - An AWS service that enables private connectivity between VPCs and services.
- **IaC**: Infrastructure as Code - The practice of managing and provisioning computing infrastructure through machine-readable definition files.

3.2 Related Documentation

- AWS PrivateLink Overview in Confluent Cloud
- Use AWS PrivateLink for Serverless Products on Confluent Cloud
- GitHub Sample Project for Confluent Terraform Provider PrivateLink Attachment
- Geo-replication with Cluster Linking on Confluent Cloud
- Use the Confluent Cloud Console with Private Networking
- IP Filtering on Confluent Cloud
- AWS/Azure PrivateLink Networking Course
- Hands On: Configuring a PrivateLink Cluster