Project Horizon

ZTNA and SASE Implementation Report

Project Title: Project Horizon - Unified Zero Trust Architecture
Document Type: Technical Implementation Report

Executive Summary

The objective of this solution was to design and deploy a secure, identity-driven access layer for internal applications using **Pomerium** as a reverse proxy and policy enforcement point, and **Keycloak** as the centralized OpenID Connect (OIDC) Identity Provider (IdP). This setup aligns with Zero Trust and SASE (Secure Access Service Edge) goals by enabling **strong authentication**, **per-user access control**, and **comprehensive logging** for security monitoring and detection engineering.

The deployed architecture provides Single Sign-On (SSO) protection for a demo application hosted on the IAM server. User authentication is managed through Keycloak, while Pomerium handles access enforcement and TLS termination for protected endpoints. The end-to-end flow implements secure redirection to the IdP, OIDC code exchange, and identity propagation to the upstream application, ensuring that only authorized users can access internal resources.

All authentication and proxy logs are captured in structured JSON format and forwarded to the ELK stack, enabling real-time visibility into authentication events, route decisions, and policy enforcement actions. This logging foundation supports downstream use cases including dashboarding, threat detection, and incident response.

Additionally, **pfSense** with **Suricata IDS/IPS** has been deployed on the virtual network to monitor network traffic associated with this environment. This complements the identity-centric Zero Trust enforcement with network-level inspection and alerting.

This implementation demonstrates a **fully functional SASE + ZTNA prototype** using open-source components. It satisfies core client requirements such as centralized identity, secure application publishing, TLS encryption, and actionable security telemetry. While designed for a controlled lab environment, the solution establishes a strong foundation for future scaling and integration into production environments.

1. Introduction

1.1 Requirements

Industry required a secure access solution capable of enforcing identity-based authentication, encrypted application publishing, and centralized visibility for user activity. The key requirements included:

- **Zero Trust Network Access (ZTNA):** Ensure that access to internal applications is strictly authenticated and authorized through a centralized identity provider, with no implicit trust based on network location.
- **Single Sign-On (SSO):** Provide a seamless authentication experience using a trusted OpenID Connect (OIDC)-compliant IdP.
- **TLS Encryption:** Secure all client–proxy and proxy–upstream communications using valid TLS certificates.
- **Comprehensive Logging:** Generate structured logs for all authentication and authorization events, suitable for ingestion into the ELK stack for detection engineering, dashboards, and alerting.
- **SASE Alignment:** Integrate network-level monitoring and inspection alongside identity enforcement to support a layered defense strategy.

1.2 Solution Implemented

To meet these requirements, a **Pomerium + Keycloak** stack was deployed on the IAM host to provide identity-aware reverse proxy functionality and centralized authentication.

- **Keycloak** (v21.1.1) acts as the OIDC Identity Provider (IdP), managing realms, clients, and user credentials.
- **Pomerium** (v0.30.6) functions as the secure reverse proxy and policy engine, enforcing access rules and handling TLS termination.
- A **demo web application** hosted on the IAM server was protected behind Pomerium, with user access restricted to authorized identities from Keycloak.

- Structured **JSON logs** from both Keycloak and Pomerium were configured and ingested into the ELK stack to provide real-time visibility into authentication events and policy decisions.
- **pfSense** with **Suricata IDS/IPS** was deployed on the virtual network to monitor network traffic associated with this setup, ensuring that access control is complemented by network-layer inspection.

This implementation delivers a functional and secure ZTNA prototype that aligns with SASE strategy. It demonstrates how identity-based access, encrypted communication, and security monitoring can be integrated using open-source technologies in a controlled environment.

2. Scope

2.1 Current Scope

The current implementation focuses on establishing a **functional SASE + ZTNA prototype** using open-source components within a controlled lab environment. The scope includes:

- **Single protected application** published through Pomerium, fronted by TLS, and integrated with Keycloak as the identity provider.
- **SSO enforcement** for application access, with authentication handled via OIDC and user credentials managed in Keycloak.
- **Centralized security telemetry** by enabling structured JSON logging on both Pomerium and Keycloak, with ingestion into the ELK stack for visibility, detection engineering, and dashboarding.
- **Network monitoring** through pfSense and Suricata, providing IDS/IPS inspection and forwarding relevant network telemetry to ELK.
- **Basic detection rules and dashboards** in ELK for monitoring sign-in activity and authentication events.

This scope demonstrates core Zero Trust principles: identity-aware access control, encrypted communication, centralized logging, and layered network visibility.

2.2 Constraints

The current deployment intentionally uses **open-source and free-tier tools**, which introduces several constraints:

- **Single-node environment:** All components are hosted on a single IAM server without redundancy, clustering, or load balancing.
- **Limited federation capabilities:** Keycloak is configured as a standalone IdP without upstream enterprise federation or SAML integration.
- **Non-production TLS certificates:** Certificates are self-signed and distributed via manual trust configuration; no enterprise CA or automated renewal is in place.
- **Limited policy granularity:** Access policies are applied on a per-route basis for a single test user; no advanced role-based or attribute-based controls are configured.
- **Prototype-scale telemetry:** Logging and detection are set up for demonstration purposes, without production-grade retention, enrichment, or alerting pipelines.

These constraints reflect the deliberate choice to prioritize rapid functional demonstration over full enterprise hardening.

2.3 Gaps and Future Enhancements

To transition from prototype to production readiness, the following areas have been identified for enhancement:

- **Federated Identity & SSO:** Integration with upstream enterprise identity providers (e.g., Azure AD, Okta) for scalable SSO and MFA.
- **High Availability:** Deploying Keycloak and Pomerium in redundant configurations with load balancing and session persistence.
- **Certificate Management:** Implementing automated certificate issuance and renewal through enterprise PKI or ACME.
- **Granular Policy Controls:** Expanding Pomerium policies to include group-based and context-aware rules, supporting more complex access scenarios.
- **Monitoring and Alerting:** Scaling log ingestion and analytics to handle higher volumes, with SIEM correlation rules, anomaly detection, and alert automation.

- **Integration with Broader SASE Stack:** Extending the solution to cover more applications, integrate with CASB/RBI tools, and support distributed PoPs for low-latency access.

Assumptions

This solution assumes the following conditions for correct operation:

- Client systems have appropriate host file entries or DNS resolution configured to access the IAM server endpoints.
- TLS certificates used for Keycloak and Pomerium are trusted on client systems.
- Basic network connectivity between all components is stable and unrestricted for the required ports.
- ELK stack is operational and able to ingest JSON log data from Pomerium and Keycloak.

3. Tools & Solution Overview

The SASE + ZTNA prototype integrates a set of **open-source identity, proxy, security monitoring, and logging components** to deliver secure, authenticated access to internal applications. Each tool plays a distinct role in enforcing Zero Trust principles, providing visibility, and enabling future scalability. The table and subsections below summarize the deployed components.

Component	Purpose	Key Functions
Keycloak 21.1.1	Identity Provider (IdP)	OIDC-based authentication, realm and user management, client configuration
Pomerium 0.30.6	Reverse proxy & ZTNA policy enforcement	TLS termination, identity-aware routing, access control, identity header forwarding
Demo Application	Protected upstream application	Static web app served via Python HTTP server for authentication and access testing
pfSense + Suricata	Network monitoring and IDS/IPS	Traffic inspection, network telemetry generation, VNet monitoring

ELK Stack	Centralized logging and detection	Structured JSON log ingestion, dashboards, and detection engineering for auth events

3.1 Keycloak (OIDC Identity Provider)

Keycloak serves as the **centralized identity provider** for this solution. A dedicated realm (aether-realm) was created, along with a confidential OIDC client (pomerium) configured for standard authorization code flow with PKCE.

- **Public Endpoints:** HTTPS on port 8443, hostname keycloak.aether.local.
- **User Management:** A test user (contractor1@aether.local) was created with a persistent password for authentication testing.
- **TLS:** Keycloak uses server certificates stored under /etc/keycloak/certs/. The CA certificate was imported into the IAM host trust store to ensure mutual TLS validation.
- **Logging:** JSON console logging is enabled and redirected to system log files for ingestion by ELK.

3.2 Pomerium (ZTNA Reverse Proxy)

Pomerium acts as the **ZTNA policy enforcement point**, mediating all traffic to protected applications.

- **Public Endpoints:**
    - https://app.aether.local → Protected demo application
    - https://authenticate.aether.local → Pomerium authentication UI
- **Configuration:** Defined in /home/identity/config.yaml, including IdP settings, route definitions, TLS certificate paths, and allowed users.
- **Access Control:** Only contractor1@aether.local is authorized for the demo route. Identity headers (X-Email) are forwarded to the upstream application.
- **TLS:** Pomerium terminates TLS using certificates stored at /etc/pomerium/certs/.
- **Systemd Integration:** Runs as a managed service to avoid Envoy socket conflicts and ensure consistent startup.

- **Logging:** Outputs structured JSON logs to /var/log/pomerium/pomerium.json for authentication flow visibility.

3.3 Demo Application

A static web page (/home/identity/index.html) is hosted using Python's built-in HTTP server on port 8081.

- Acts as a **representative internal service** published behind the Pomerium proxy.
- Receives identity headers after successful authentication, verifying end-to-end access enforcement.

3.4 pfSense with Suricata

A pfSense virtual appliance with Suricata IDS/IPS is deployed on the monitored VNet segment.

- **Purpose:** Observe and inspect traffic flows associated with Keycloak, Pomerium, and the demo application.
- **Integration:** Suricata-generated logs are forwarded to ELK, complementing identity telemetry with network-level visibility.
- **Relevance to SASE:** Provides the network inspection component of the layered security model.

3.5 ELK Stack (Logging and Detection)

All authentication and proxy logs are generated in structured JSON format and ingested into the ELK stack for centralized analysis.

- **Sources:**
    - Keycloak logs: /var/log/keycloak/keycloak.json
    - Pomerium logs: /var/log/pomerium/pomerium.json
- **Dashboards & Rules:** Authentication dashboards and sign-in detection rules are configured in ELK for monitoring ZTNA activity.
- **Extensibility:** Log forwarding is set up to support future correlation, anomaly detection, and security automation use cases.

This toolchain collectively enables secure identity enforcement, encrypted access, telemetry generation, and centralized monitoring. It forms a **cohesive open-source foundation** for demonstrating SASE + ZTNA principles in a controlled environment, while remaining extensible for future scaling and production integration.

4. Configurations & Customizations

The deployed solution involved several **non-default configurations and targeted customizations** across Keycloak, Pomerium, TLS handling, logging, and system services to ensure secure, identity-aware access and proper telemetry generation. These changes were essential to achieve end-to-end authentication, routing, and monitoring in line with the client's SASE and ZTNA objectives.

4.1 Keycloak Configuration

The Keycloak identity provider was configured to enable OIDC-based SSO for the demo application protected by Pomerium.

- **Realm Configuration**
    - A new realm named aether-realm was created and enabled.
    - HTTPS endpoints were served on port 8443, with the hostname keycloak.aether.local.
- **Client Configuration**
    - A confidential OIDC client named pomerium was created for Pomerium integration.
    - **Redirect URIs:**
        - https://authenticate.aether.local/*
        - https://authenticate.aether.local/oauth2/callback
    - **Web Origins:**
        - https://authenticate.aether.local
    - Standard Authorization Code flow was enabled, along with Direct Access Grants for testing.

    - PKCE (S256) was retained as default for secure code exchange.
    - The client secret was generated and securely inserted into the Pomerium configuration.
- **User Configuration**
    - A test user contractor1@aether.local was created with a non-temporary password for end-to-end authentication testing.
- **TLS Trust Configuration**
    - The Keycloak CA certificate was copied to /usr/local/share/ca-certificates/keycloak.crt and added to the system trust store using update-ca-certificates.
    - This resolved OIDC x509 authority validation errors between Pomerium and Keycloak.
- **Logging**
    - JSON console logging was enabled via the environment variable:

    ```
    QUARKUS_LOG_CONSOLE_JSON=true
    ```

    - Systemd unit output was redirected to structured log files for ingestion:
        - /var/log/keycloak/keycloak.json
        - /var/log/keycloak/keycloak.err.json

4.2 Pomerium Configuration

Pomerium was configured as the reverse proxy and ZTNA enforcement point through /home/identity/config.yaml.

- **Identity Provider Settings**

```
idp_provider: oidc

idp_provider_url: https://keycloak.aether.local:8443/realms/aether-realm

idp_client_id: pomerium

idp_client_secret: "<client-secret>"
```

```
idp_scopes: ["openid","email","profile"]
```

- **TLS Certificates**

```
certificates:
  - cert: /etc/pomerium/certs/pomerium.crt
    key:  /etc/pomerium/certs/pomerium.key
```

- **Route & Access Policy**

```
routes:
  - from: https://app.aether.local
    to: http://localhost:8081
    allowed_users:
      - contractor1@aether.local
```

  - Only authenticated users matching the allowed list are granted access.
  - Pomerium forwards the X-Email header to the upstream application to propagate identity context.

- **Binary & Service Path Adjustments**
  - The binary path was corrected from /usr/local/bin/pomerium to /usr/sbin/pomerium.
  - Stray manual runs were terminated to prevent Envoy socket binding conflicts, and only the systemd-managed instance is allowed to run.
- **Logging**
  - Pomerium outputs structured JSON logs by default. These were redirected to persistent files:
    - /var/log/pomerium/pomerium.json
    - /var/log/pomerium/pomerium.err.json

4.3 TLS & Hostname Customizations

- Host entries were added on client machines to resolve the IAM server’s public IP:

  keycloak.aether.local

  authenticate.aether.local

  app.aether.local

- Pomerium certificates were placed under /etc/pomerium/certs/.
- Keycloak certificates were placed under /etc/keycloak/certs/.
- Self-signed certificates were trusted locally to enable secure TLS without commercial CA dependencies.

4.4 Systemd Services

- **Keycloak Service**
  - Defined in /etc/systemd/system/keycloak.service.
  - Uses kc.sh start-dev with HTTPS on 8443 and explicit hostname binding.
- **Pomerium Service**
  - Defined in /etc/systemd/system/pomerium.service.
  - Runs using:

    ```
    ExecStart=/usr/sbin/pomerium --config /home/identity/config.yaml
    ```
  - Systemd control resolved earlier conflicts with manually launched instances.

4.5 Logging and Ingestion Pipeline

- **Log Rotation**
  - Weekly rotation and compression were configured to prevent uncontrolled file growth (rotate 8 copies).
- **ELK Ingestion**
  - All log files from Keycloak and Pomerium are ingested into the ELK stack for authentication monitoring and detection engineering.

- Example detection: Sign-in events from Pomerium are monitored to trigger alerts and populate dashboards.

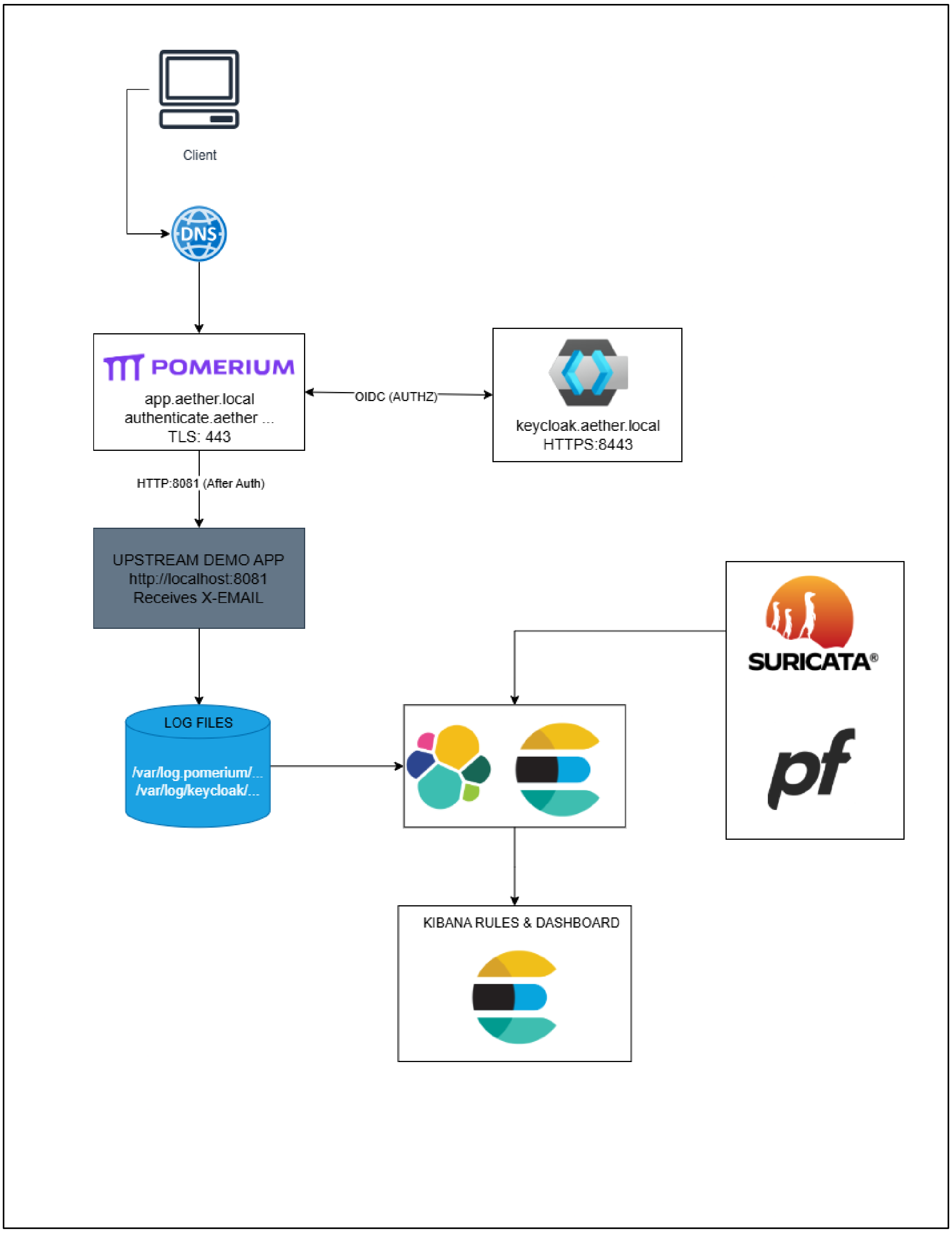
These targeted configurations and deviations from default behavior were critical to achieving a **secure, auditable, and identity-aware access control layer** using open-source technologies. Each change was purposeful either to enable interoperability, enforce security controls, or support downstream monitoring.

5. Architecture & Data Flow Diagram

5.1 High-Level View

The solution implements identity-aware access to an internal web application by placing **Pomerium** (ZTNA reverse proxy and policy engine) in front of the upstream app and integrating it with **Keycloak** (OIDC IdP). All security-relevant events are logged in **structured JSON** and ingested into **ELK**. **pfSense with Suricata** observes network traffic on the monitored VNet to complement identity telemetry.

5.2 Diagram



5.3 Trust Boundaries & Identities

- **Public/TLS Boundary:** Client ↔ Pomerium (app.aether.local, authenticate.aether.local) over HTTPS (TLS served by Pomerium).

- **IdP Boundary:** Pomerium ↔ Keycloak over HTTPS (Keycloak CA imported into system trust on IAM).
- **Backend Boundary:** Pomerium ↔ Upstream demo app over local HTTP (loopback/host-local). Access allowed **only after** successful OIDC flow and policy evaluation.
- **Telemetry Boundary:** Pomerium and Keycloak emit **JSON logs** locally; ELK ingests these for dashboards and detections. pfSense/Suricata contributes network-level events.

5.4 End-to-End Data Flow (happy path)

1. **Initial Request:**
   Client requests https://app.aether.local/ (TLS terminated by Pomerium).
2. **Policy Gate / Redirect:**
   Pomerium evaluates route policy. No session → redirect to https://authenticate.aether.local/.pomerium/sign_in.
3. **OIDC Authorization (Code Flow):**
   Pomerium initiates OIDC with Keycloak (aether-realm, client pomerium), scopes openid email profile.
4. **User Authentication:**
   User signs in on Keycloak (e.g., contractor1@aether.local). Keycloak returns an **authorization code** to Pomerium's callback.
5. **Token Exchange & Session:**
   Pomerium exchanges the code for tokens, establishes a session, and applies policy (allowed_users list).
6. **Upstream Request (Authorized):**
   Pomerium forwards the request to http://localhost:8081, injecting **X-Email: contractor1@aether.local** to propagate identity.
7. **Response to Client:**
   Client receives the protected content from the upstream app via Pomerium.
8. **Telemetry:**

  - Pomerium logs: redirects, OIDC events, policy decisions → /var/log/pomerium/pomerium.json.
  - Keycloak logs: authentication events → /var/log/keycloak/keycloak.json.
  - pfSense/Suricata: network observations from the VNet.
  - ELK ingests these for dashboards and detection use-cases (e.g., sign-in visibility).

5.5 Error/Exception Paths (Selected)

- **Unauthorized User:** Policy denies access → Pomerium returns appropriate 403 and logs the decision.
- **TLS Trust Issue (IdP):** If Keycloak CA is not trusted, OIDC back-channel fails; Pomerium logs x509 errors.
- **Misconfigured Redirect URIs:** OIDC callback mismatch yields invalid_request/CSRF errors; visible in Pomerium and Keycloak logs.

5.6 Identity & Authorization Model

- **Authentication:** Keycloak (OIDC Authorization Code with PKCE S256).
- **Authorization (ZTNA):** Pomerium enforces **per-route** policy (allowed_users), with identity propagated to the app via X-Email.
- **Session:** Maintained by Pomerium after successful OIDC exchange.

5.7 Observability & Logging

- **Format:** Structured JSON for machine readability and SIEM correlation.
- **Locations:**
  - Pomerium: /var/log/pomerium/pomerium.json, /var/log/pomerium/pomerium.err.json
  - Keycloak: /var/log/keycloak/keycloak.json, /var/log/keycloak/keycloak.err.json
- **Lifecycle:** Logrotate configured (weekly, rotate 8, compress) to control growth.

- **Consumption:** Shipped to ELK for dashboards and detection engineering (e.g., authentication activity, ZTNA access patterns).

5.8 Network, Ports & Certificates (Summary)

- **Client → Pomerium:** HTTPS :443 (app.aether.local, authenticate.aether.local) using /etc/pomerium/certs/pomerium.{crt,key}.
- **Pomerium → Keycloak:** HTTPS :8443 (keycloak.aether.local) using Keycloak's server cert; CA trusted via system store.
- **Pomerium → Upstream App:** HTTP :8081 (local host).
- **pfSense/Suricata:** Monitors VNet interfaces; forwards events to ELK.

5.9 Deployment Topology (Operational Notes)

- **Systemd-managed services** for both Pomerium and Keycloak prevent Envoy socket conflicts and ensure deterministic startup.
- **Host/DNS entries** on client systems resolve all *.aether.local hostnames to the IAM server.
- **Certificates** are self-signed for the prototype; trust is explicitly established on participating systems.

6. Evidence & PoC

The implementation of the SASE + ZTNA solution was validated through a series of functional tests, service verifications, authentication flows, and log inspections. The goal of this PoC was to demonstrate **secure identity-based access enforcement**, **end-to-end authentication flows**, and **visibility through centralized logging**. All evidence was collected from the lab environment to substantiate the deployment and configuration steps.

6.1 Service Verification

Before initiating authentication tests, all core services were validated to ensure proper startup and availability:

- **Keycloak**

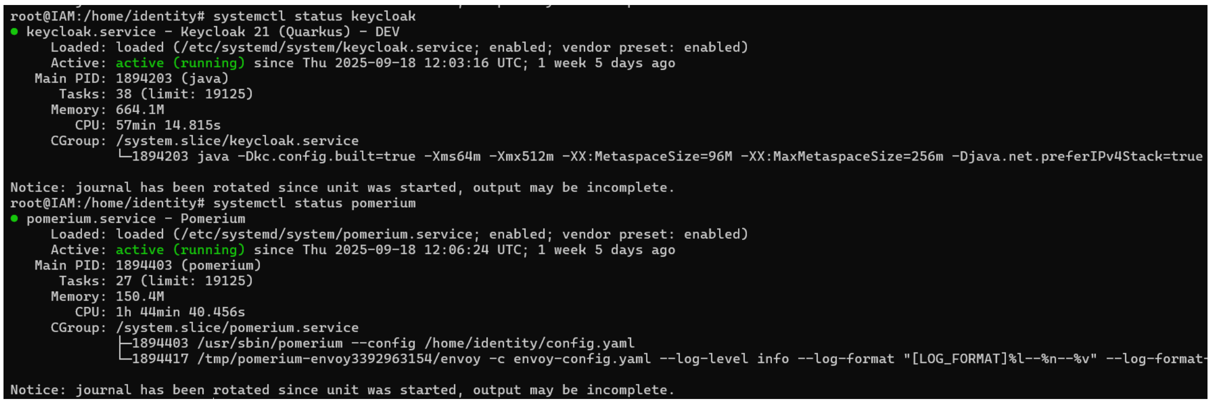
```
systemctl status keycloak
```

Verified that Keycloak is running in development mode with HTTPS on port 8443 and accessible via https://keycloak.aether.local/.

- **Pomerium**

  ```
  systemctl status pomerium
  ```

Confirmed that Pomerium is running as a systemd service with correct binary path (/usr/sbin/pomerium) and valid configuration at /home/identity/config.yaml.



```
root@IAM:/home/identity# systemctl status keycloak
● keycloak.service - Keycloak 21 (Quarkus) - DEV
     Loaded: loaded (/etc/systemd/system/keycloak.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2025-09-18 12:03:16 UTC; 1 week 5 days ago
   Main PID: 1894203 (java)
      Tasks: 38 (limit: 19125)
     Memory: 664.1M
        CPU: 57min 14.815s
     CGroup: /system.slice/keycloak.service
             └─1894203 java -Dkc.config.built=true -Xms64m -Xmx512m -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256m -Djava.net.preferIPv4Stack=true

Notice: journal has been rotated since unit was started, output may be incomplete.
root@IAM:/home/identity# systemctl status pomerium
● pomerium.service - Pomerium
     Loaded: loaded (/etc/systemd/system/pomerium.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2025-09-18 12:06:24 UTC; 1 week 5 days ago
   Main PID: 1894403 (pomerium)
      Tasks: 27 (limit: 19125)
     Memory: 150.4M
        CPU: 1h 44min 40.456s
     CGroup: /system.slice/pomerium.service
             ├─1894403 /usr/sbin/pomerium --config /home/identity/config.yaml
             └─1894417 /tmp/pomerium-envoy3392963154/envoy -c envoy-config.yaml --log-level info --log-format "[LOG_FORMAT]%l--%n--%v" --log-format-

Notice: journal has been rotated since unit was started, output may be incomplete.
```

- **Demo Application**

  ```
  python3 -m http.server 8081 --bind 0.0.0.0
  ```

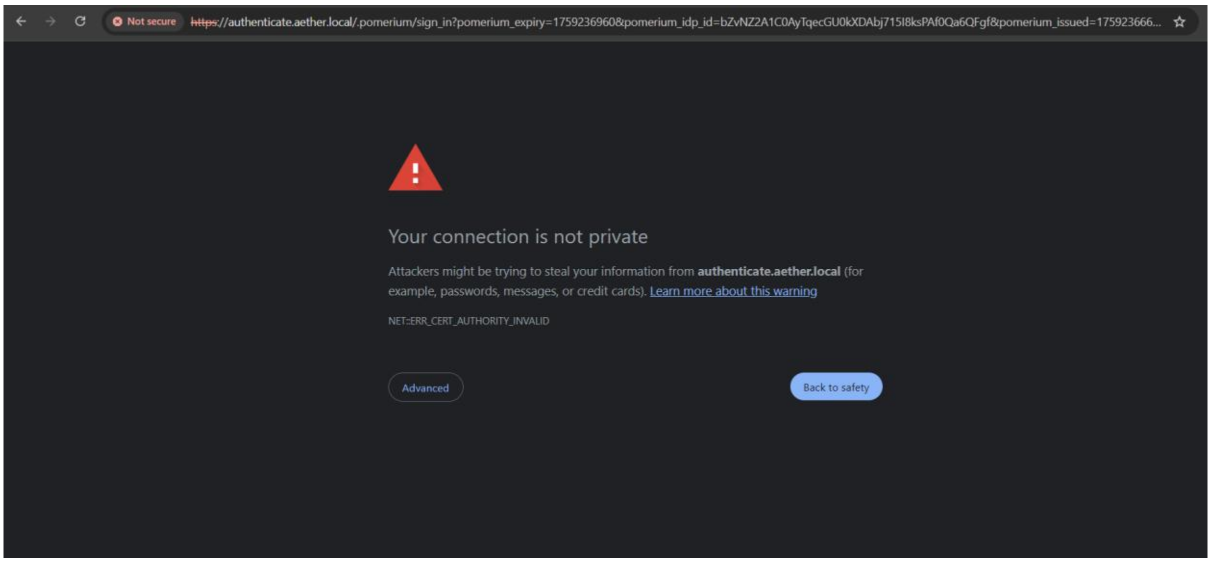
Ensured the static application is accessible locally on port 8081 and correctly serving /home/identity/index.html.

Successful service verification established the baseline for the SSO authentication flow.

6.2 End-to-End Authentication Flow

The **happy path** login flow was tested from a client workstation configured with the appropriate hosts entries pointing to the IAM server:

1. **Initial Access:**
   User navigated to https://app.aether.local/.
   → Redirected by Pomerium to https://authenticate.aether.local/.pomerium/sign_in.


Not secure ~~https~~://authenticate.aether.local/.pomerium/sign_in?pomerium_expiry=1759236960&pomerium_idp_id=bZvNZ2A1C0AyTqecGU0kXDAbj715I8ksPAf0Qa6QFgf&pomerium_issued=175923666...

Your connection is not private

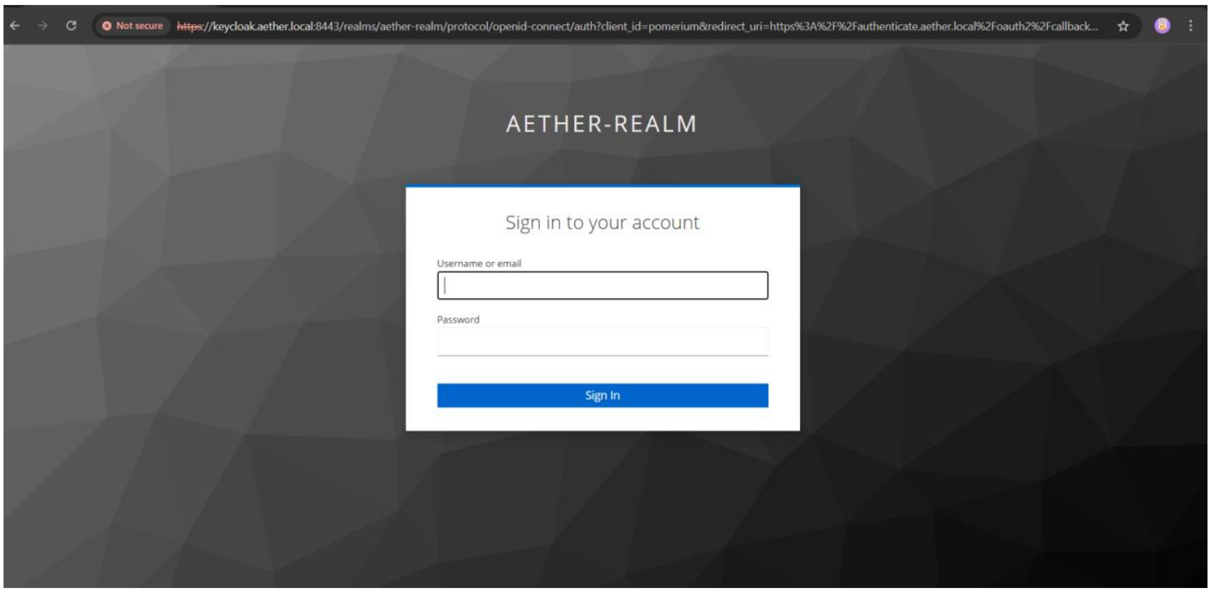
Attackers might be trying to steal your information from **authenticate.aether.local** (for example, passwords, messages, or credit cards). Learn more about this warning

NET::ERR_CERT_AUTHORITY_INVALID

2. **Identity Provider Redirection:**

   Pomerium initiated OIDC flow with Keycloak (aether-realm).

   → Keycloak login page displayed over HTTPS (8443).


Not secure ~~https~~://keycloak.aether.local:8443/realms/aether-realm/protocol/openid-connect/auth?client_id=pomerium&redirect_uri=https%3A%2F%2Fauthenticate.aether.local%2Foauth2%2Fcallback...

AETHER-REALM

Sign in to your account

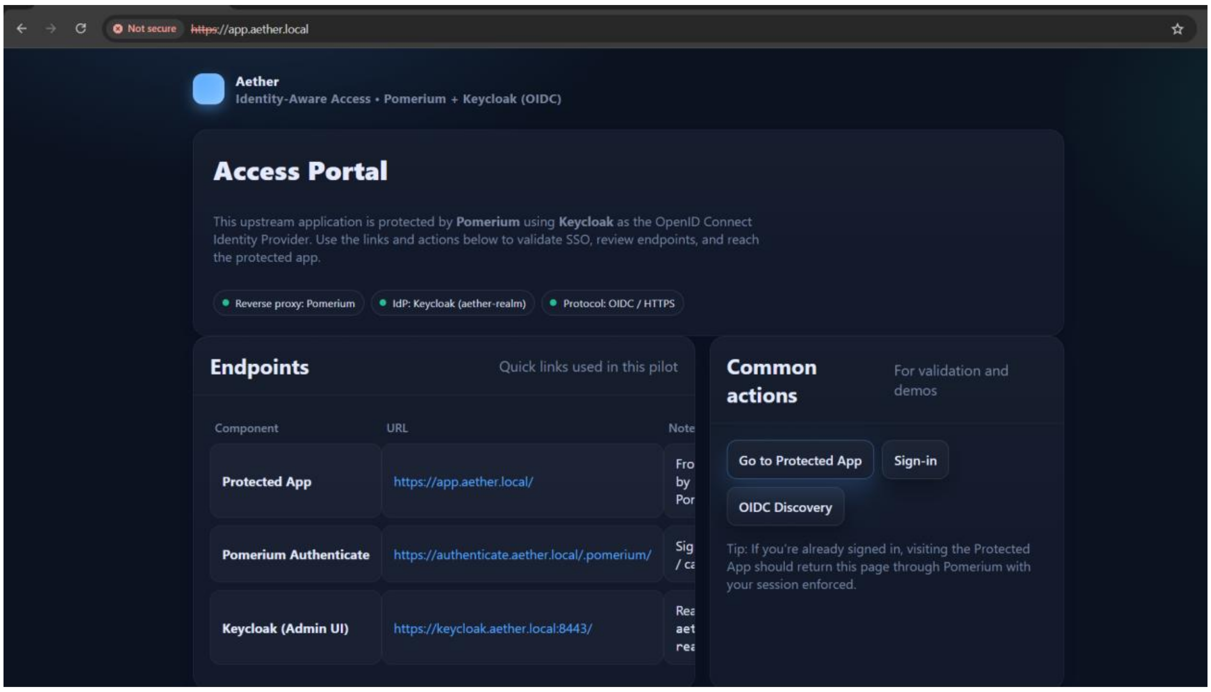
Username or email

Password

3. **User Authentication:**

   Logged in using contractor1@aether.local credentials.

   → Successful authorization code issued to Pomerium callback endpoint.


Not secure ~~https~~://app.aether.local

Aether
Identity-Aware Access • Pomerium + Keycloak (OIDC)

Access Portal

This upstream application is protected by **Pomerium** using **Keycloak** as the OpenID Connect Identity Provider. Use the links and actions below to validate SSO, review endpoints, and reach the protected app.

● Reverse proxy: Pomerium ● IdP: Keycloak (aether-realm) ● Protocol: OIDC / HTTPS

Endpoints Quick links used in this pilot

Component	URL	Note
Protected App	https://app.aether.local/	Fro by Por
Pomerium Authenticate	https://authenticate.aether.local/.pomerium/	Sig / ca
Keycloak (Admin UI)	https://keycloak.aether.local:8443/	Rea aet rea

Common actions For validation and demos

Tip: If you're already signed in, visiting the Protected App should return this page through Pomerium with your session enforced.

4. **Access Enforcement:**
   Pomerium verified the token and allowed access based on the allowed_users route policy.
   → Demo app content displayed; upstream received X-Email: contractor1@aether.local.

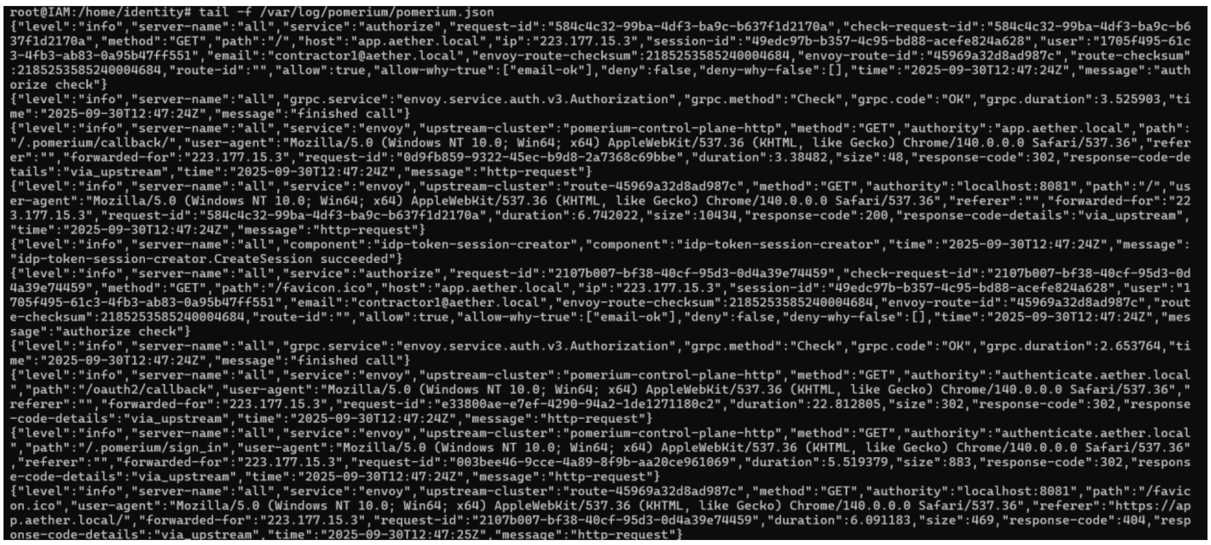
5. **Session Maintenance:**
   Subsequent requests bypassed authentication until session expiration.

This flow confirmed that ZTNA policy enforcement, OIDC integration, and TLS configuration are working correctly.

6.3 Logging & Observability

Structured JSON logs were generated by both Pomerium and Keycloak during authentication flows.

- **Pomerium Logs**
  Located at /var/log/pomerium/pomerium.json.
  Contain redirect events, OIDC handshake steps, policy decisions, and upstream request forwarding.

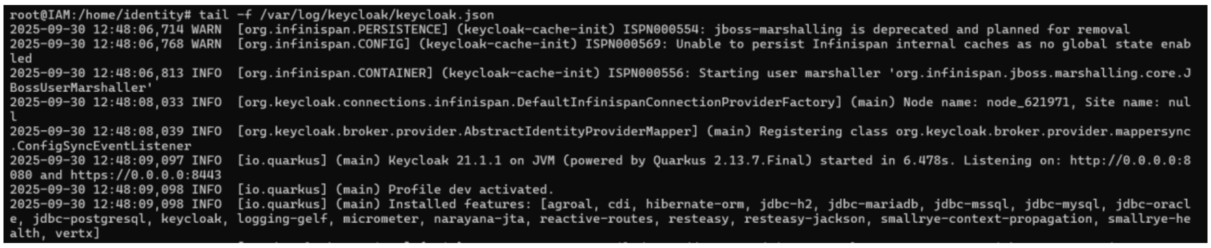


```
root@IAM:/home/identity# tail -f /var/log/pomerium/pomerium.json
{"level":"info","server-name":"all","service":"authorize","request-id":"584c4c32-99ba-4df3-ba9c-b637f1d2170a","check-request-id":"584c4c32-99ba-4df3-ba9c-b6
37f1d2170a","method":"GET","path":"/","host":"app.aether.local","ip":"223.177.15.3","session-id":"49edc97b-b357-4c95-bd88-acefe824a628","user":"1705f495-61c
3-4fb3-ab83-0a95b47ff551","email":"contractor1@aether.local","envoy-route-checksum":2185253585240004684,"envoy-route-id":"45969a32d8ad987c","route-checksum"
:2185253585240004684,"route-id":"","allow":true,"allow-why-true":["email-ok"],"deny":false,"deny-why-false":[],"time":"2025-09-30T12:47:24Z","message":"auth
orize check"}
{"level":"info","server-name":"all","grpc.service":"envoy.service.auth.v3.Authorization","grpc.method":"Check","grpc.code":"OK","grpc.duration":3.525903,"ti
me":"2025-09-30T12:47:24Z","message":"finished call"}
{"level":"info","server-name":"all","service":"envoy","upstream-cluster":"pomerium-control-plane-http","method":"GET","authority":"app.aether.local","path":
"/.pomerium/callback/","user-agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36","refer
er":"","forwarded-for":"223.177.15.3","request-id":"0d9fb859-9322-45ec-b9d8-2a7368c69bbe","duration":3.38482,"size":48,"response-code":302,"response-code-de
tails":"via_upstream","time":"2025-09-30T12:47:24Z","message":"http-request"}
{"level":"info","server-name":"all","service":"envoy","upstream-cluster":"route-45969a32d8ad987c","method":"GET","authority":"localhost:8081","path":"/","us
er-agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36","referer":"","forwarded-for":"22
3.177.15.3","request-id":"584c4c32-99ba-4df3-ba9c-b637f1d2170a","duration":6.742022,"size":10434,"response-code":200,"response-code-details":"via_upstream",
"time":"2025-09-30T12:47:24Z","message":"http-request"}
{"level":"info","server-name":"all","component":"idp-token-session-creator","component":"idp-token-session-creator","time":"2025-09-30T12:47:24Z","message":
"idp-token-session-creator.CreateSession succeeded"}
{"level":"info","server-name":"all","service":"authorize","request-id":"2107b007-bf38-40cf-95d3-0d4a39e74459","check-request-id":"2107b007-bf38-40cf-95d3-0d
4a39e74459","method":"GET","path":"/favicon.ico","host":"app.aether.local","ip":"223.177.15.3","session-id":"49edc97b-b357-4c95-bd88-acefe824a628","user":"1
705f495-61c3-4fb3-ab83-0a95b47ff551","email":"contractor1@aether.local","envoy-route-checksum":2185253585240004684,"envoy-route-id":"45969a32d8ad987c","rout
e-checksum":2185253585240004684,"route-id":"","allow":true,"allow-why-true":["email-ok"],"deny":false,"deny-why-false":[],"time":"2025-09-30T12:47:24Z","mes
sage":"authorize check"}
{"level":"info","server-name":"all","grpc.service":"envoy.service.auth.v3.Authorization","grpc.method":"Check","grpc.code":"OK","grpc.duration":2.653764,"ti
me":"2025-09-30T12:47:24Z","message":"finished call"}
{"level":"info","server-name":"all","service":"envoy","upstream-cluster":"pomerium-control-plane-http","method":"GET","authority":"authenticate.aether.local
","path":"/oauth2/callback","user-agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36","
referer":"","forwarded-for":"223.177.15.3","request-id":"e33800ae-e7ef-4290-94a2-1de1271180c2","duration":22.812805,"size":302,"response-code":302,"response
-code-details":"via_upstream","time":"2025-09-30T12:47:24Z","message":"http-request"}
{"level":"info","server-name":"all","service":"envoy","upstream-cluster":"pomerium-control-plane-http","method":"GET","authority":"authenticate.aether.local
","path":"/.pomerium/sign_in","user-agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36"
,"referer":"","forwarded-for":"223.177.15.3","request-id":"003bee46-9cce-4a89-8f9b-aa20ce961069","duration":5.519379,"size":883,"response-code":302,"respons
e-code-details":"via_upstream","time":"2025-09-30T12:47:24Z","message":"http-request"}
{"level":"info","server-name":"all","service":"envoy","upstream-cluster":"route-45969a32d8ad987c","method":"GET","authority":"localhost:8081","path":"/favic
on.ico","user-agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/140.0.0.0 Safari/537.36","referer":"https://ap
p.aether.local/","forwarded-for":"223.177.15.3","request-id":"2107b007-bf38-40cf-95d3-0d4a39e74459","duration":6.091183,"size":469,"response-code":404,"resp
onse-code-details":"via_upstream","time":"2025-09-30T12:47:25Z","message":"http-request"}
```

- **Keycloak Logs**

  Located at /var/log/keycloak/keycloak.json.

  Contain login attempts, realm selection, token issuance, and OIDC events.



```
root@IAM:/home/identity# tail -f /var/log/keycloak/keycloak.json
2025-09-30 12:48:06,714 WARN  [org.infinispan.PERSISTENCE] (keycloak-cache-init) ISPN000554: jboss-marshalling is deprecated and planned for removal
2025-09-30 12:48:06,768 WARN  [org.infinispan.CONFIG] (keycloak-cache-init) ISPN000569: Unable to persist Infinispan internal caches as no global state enab
led
2025-09-30 12:48:06,813 INFO  [org.infinispan.CONTAINER] (keycloak-cache-init) ISPN000556: Starting user marshaller 'org.infinispan.jboss.marshalling.core.J
BossUserMarshaller'
2025-09-30 12:48:08,033 INFO  [org.keycloak.connections.infinispan.DefaultInfinispanConnectionProviderFactory] (main) Node name: node_621971, Site name: nul
l
2025-09-30 12:48:08,039 INFO  [org.keycloak.broker.provider.AbstractIdentityProviderMapper] (main) Registering class org.keycloak.broker.provider.mappersync
.ConfigSyncEventListener
2025-09-30 12:48:09,097 INFO  [io.quarkus] (main) Keycloak 21.1.1 on JVM (powered by Quarkus 2.13.7.Final) started in 6.478s. Listening on: http://0.0.0.0:8
080 and https://0.0.0.0:8443
2025-09-30 12:48:09,098 INFO  [io.quarkus] (main) Profile dev activated.
2025-09-30 12:48:09,098 INFO  [io.quarkus] (main) Installed features: [agroal, cdi, hibernate-orm, jdbc-h2, jdbc-mariadb, jdbc-mssql, jdbc-mysql, jdbc-oracl
e, jdbc-postgresql, keycloak, logging-gelf, micrometer, narayana-jta, reactive-routes, resteasy, resteasy-jackson, smallrye-context-propagation, smallrye-he
alth, vertx]
```

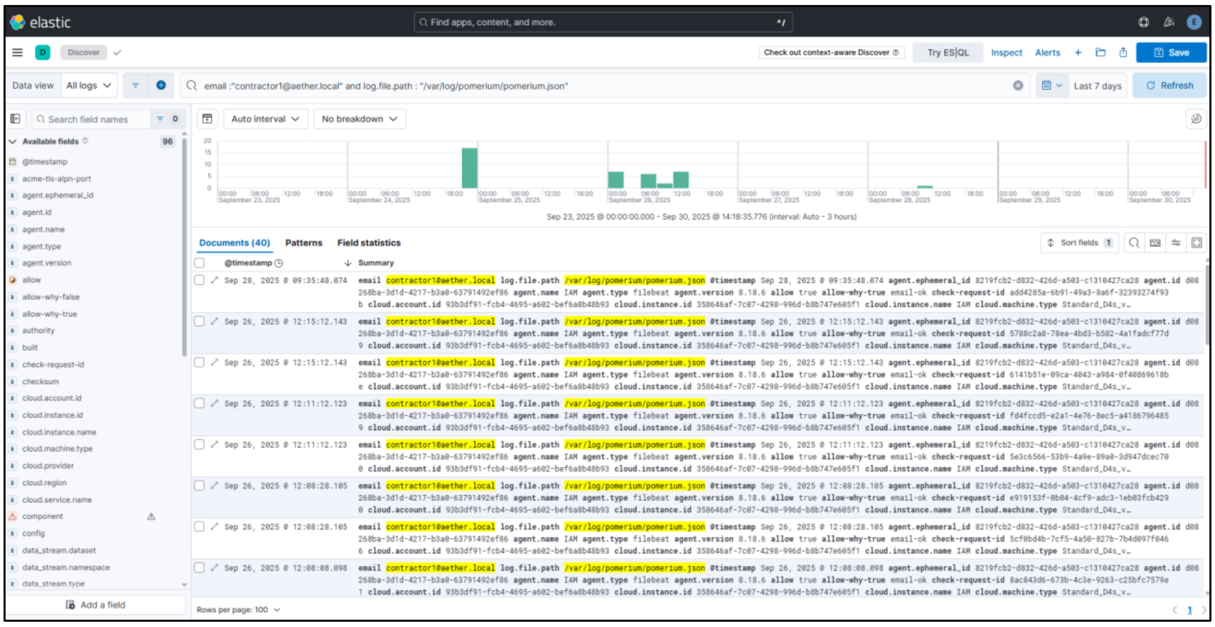
These logs were **ingested into ELK**, enabling real-time dashboards for authentication activity and detection rules for abnormal patterns.

6.4 ELK Integration & Detection

All log files were shipped to ELK through the configured forwarding pipeline. Detection rules and dashboards were created for:

- **Pomerium Sign-In Events:**

  Trigger alerts on successful authentication events to protected applications.



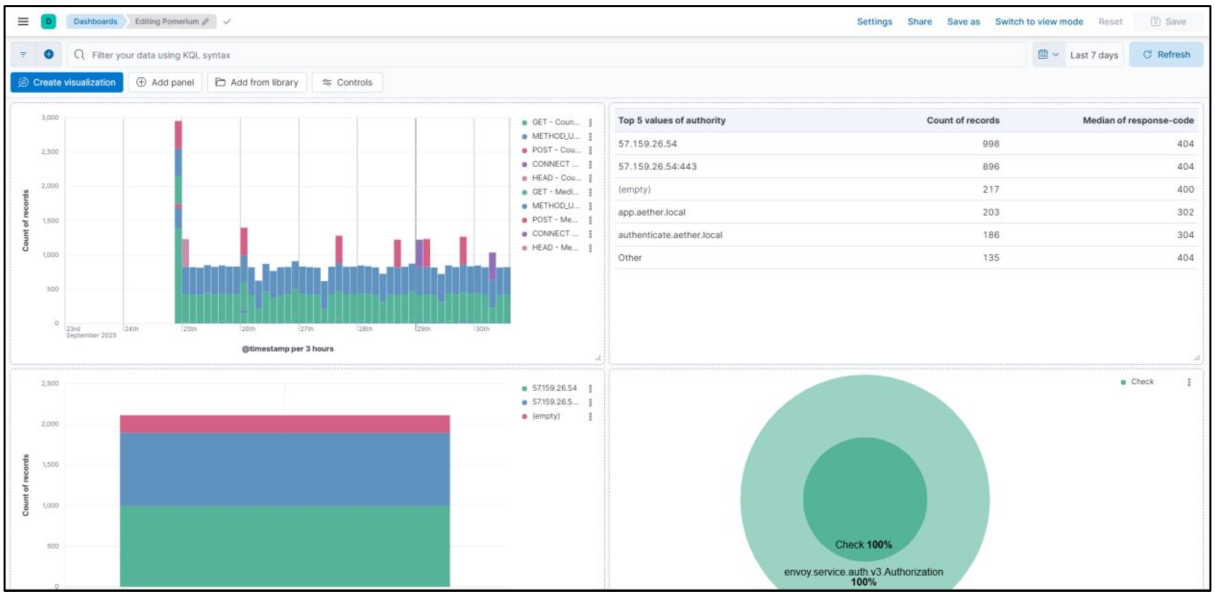
```
email :"contractor1@aether.local" and log.file.path : "/var/log/pomerium/pomerium.json"
```

Sep 23, 2025 @ 00:00:00.000 - Sep 30, 2025 @ 14:18:35.776 (interval: Auto - 3 hours)

```
@timestamp                  Summary
Sep 28, 2025 @ 09:35:48.074 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 28, 2025 @ 09:35:48.074 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id add4285a-6b91-49a3-8a6f-32393274f93
                            b cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
Sep 26, 2025 @ 12:15:12.143 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 26, 2025 @ 12:15:12.143 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id 5788c2a8-78ea-4bd3-b582-4a1fadcf77d
                            9 cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
Sep 26, 2025 @ 12:15:12.143 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 26, 2025 @ 12:15:12.143 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id 6141b51e-09ca-4043-a984-0f40869618b
                            e cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
Sep 26, 2025 @ 12:11:12.123 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 26, 2025 @ 12:11:12.123 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id fd4fccd5-e2a1-4e76-8ec5-a418679648
                            59 cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
Sep 26, 2025 @ 12:11:12.123 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 26, 2025 @ 12:11:12.123 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id 5e3c6566-53b9-4a9e-89a0-3d947dcec70
                            0 cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
Sep 26, 2025 @ 12:08:28.105 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 26, 2025 @ 12:08:28.105 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id e919153f-0b04-4cf9-adc3-1eb03fcb429
                            0 cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
Sep 26, 2025 @ 12:08:28.105 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 26, 2025 @ 12:08:28.105 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id 5cf0bd4b-7cf5-4a50-827b-7b4d097f046
                            6 cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
Sep 26, 2025 @ 12:08:08.098 email contractor1@aether.local log.file.path /var/log/pomerium/pomerium.json @timestamp Sep 26, 2025 @ 12:08:08.098 agent.ephemeral_id 8219fcb2-d832-426d-a503-c1310427ca28 agent.id d08
                            268ba-3d1d-4217-b3a0-63791492ef86 agent.name IAM agent.type filebeat agent.version 8.18.6 allow true allow-why-true email-ok check-request-id 8ac843d6-673b-4c3e-9263-c25bfc7579e
                            1 cloud.account.id 93b3df91-fcb4-4695-a602-bef6a8b48b93 cloud.instance.id 358646af-7c07-4298-996d-b8b747e605f1 cloud.instance.name IAM cloud.machine.type Standard_D4s_v…
```

- **Authentication Dashboards:**

  Visualize login trends, source IPs, users, and application access patterns.

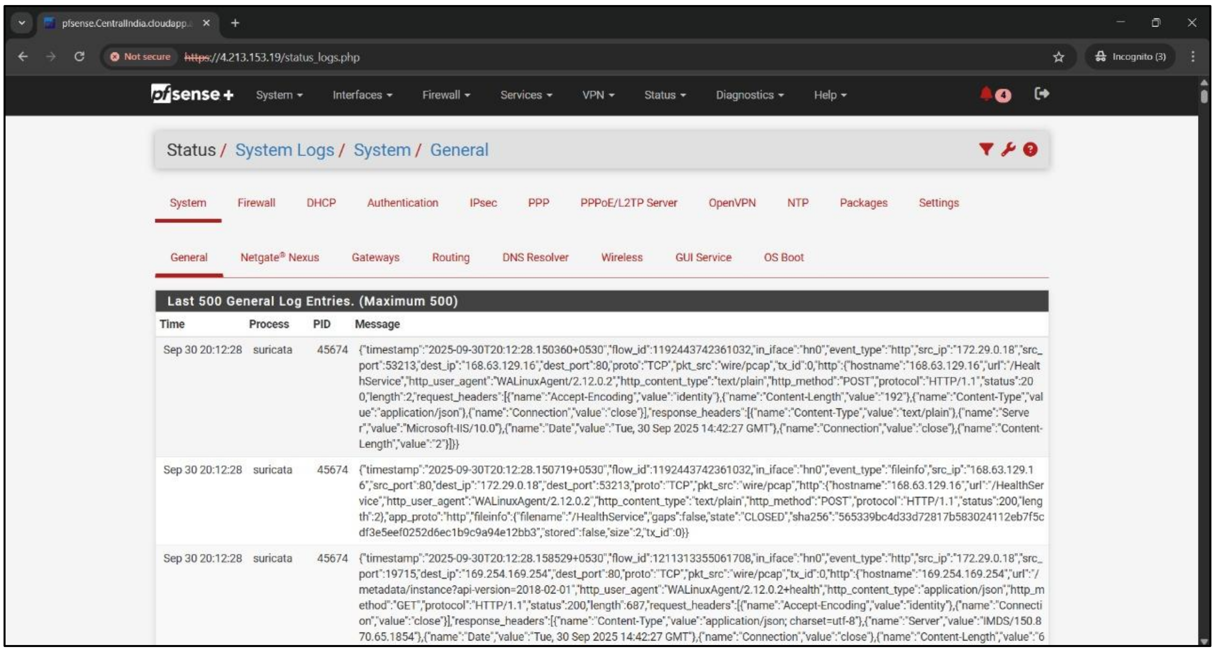


Top 5 values of authority	Count of records	Median of response-code
57.159.26.54	998	404
57.159.26.54:443	896	404
(empty)	217	400
app.aether.local	203	302
authenticate.aether.local	186	304
Other	135	404

These integrations provide **continuous visibility** into authentication activity and form the basis for detection engineering in line with Zero Trust principles.

6.5 Network Monitoring Evidence

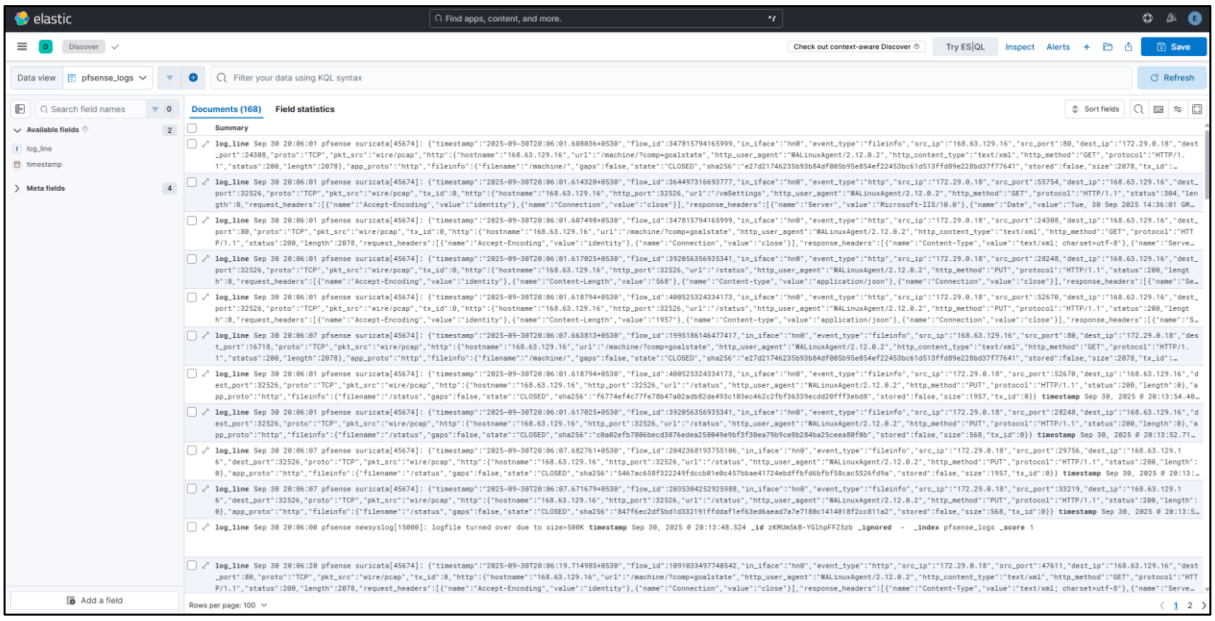
pfSense with **Suricata IDS/IPS** was deployed on the monitored virtual network.



Last 500 General Log Entries. (Maximum 500)

```
Time            Process   PID    Message
Sep 30 20:12:28 suricata  45674  {"timestamp":"2025-09-30T20:12:28.150360+0530","flow_id":1192443742361032,"in_iface":"hn0","event_type":"http","src_ip":"172.29.0.18","src_
                                 port":53213,"dest_ip":"168.63.129.16","dest_port":80,"proto":"TCP","pkt_src":"wire/pcap","tx_id":0,"http":{"hostname":"168.63.129.16","url":"/Healt
                                 hService","http_user_agent":"WALinuxAgent/2.12.0.2","http_content_type":"text/plain","http_method":"POST","protocol":"HTTP/1.1","status":20
                                 0,"length":2,"request_headers":[{"name":"Accept-Encoding","value":"identity"},{"name":"Content-Length","value":"192"},{"name":"Content-Type","val
                                 ue":"application/json"},{"name":"Connection","value":"close"}],"response_headers":[{"name":"Content-Type","value":"text/plain"},{"name":"Serve
                                 r","value":"Microsoft-IIS/10.0"},{"name":"Date","value":"Tue, 30 Sep 2025 14:42:27 GMT"},{"name":"Connection","value":"close"},{"name":"Content-
                                 Length","value":"2"}]}}
Sep 30 20:12:28 suricata  45674  {"timestamp":"2025-09-30T20:12:28.150719+0530","flow_id":1192443742361032,"in_iface":"hn0","event_type":"fileinfo","src_ip":"168.63.129.1
                                 6","src_port":80,"dest_ip":"172.29.0.18","dest_port":53213,"proto":"TCP","pkt_src":"wire/pcap","http":{"hostname":"168.63.129.16","url":"/HealthSer
                                 vice","http_user_agent":"WALinuxAgent/2.12.0.2","http_content_type":"text/plain","http_method":"POST","protocol":"HTTP/1.1","status":200,"leng
                                 th":2},"app_proto":"http","fileinfo":{"filename":"/HealthService","gaps":false,"state":"CLOSED","sha256":"565339bc4d33d72817b583024112eb7f5c
                                 df3e5eef0252d6ec1b9c9a94e12bb3","stored":false,"size":2,"tx_id":0}}
Sep 30 20:12:28 suricata  45674  {"timestamp":"2025-09-30T20:12:28.158529+0530","flow_id":1211313355061708,"in_iface":"hn0","event_type":"http","src_ip":"172.29.0.18","src_
                                 port":19715,"dest_ip":"169.254.169.254","dest_port":80,"proto":"TCP","pkt_src":"wire/pcap","tx_id":0,"http":{"hostname":"169.254.169.254","url":"/
                                 metadata/instance?api-version=2018-02-01","http_user_agent":"WALinuxAgent/2.12.0.2+health","http_content_type":"application/json","http_m
                                 ethod":"GET","protocol":"HTTP/1.1","status":200,"length":687,"request_headers":[{"name":"Accept-Encoding","value":"identity"},{"name":"Connecti
                                 on","value":"close"}],"response_headers":[{"name":"Content-Type","value":"application/json; charset=utf-8"},{"name":"Server","value":"IMDS/150.8
                                 70.65.1854"},{"name":"Date","value":"Tue, 30 Sep 2025 14:42:27 GMT"},{"name":"Connection","value":"close"},{"name":"Content-Length","value":"6
```

- Generated network telemetry for ELK.
- Validated visibility of HTTPS handshakes and allowed sessions through VNet inspection.



```
log_line Sep 30 20:06:01 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:01.608036+0530","flow_id":347815794165999,"in_iface":"hn0","event_type":"fileinfo","src_ip":"168.63.129.16","src_port":80,"dest_ip":"172.29.0.18","dest
_port":24308,"proto":"TCP","pkt_src":"wire/pcap","http":{"hostname":"168.63.129.16","url":"/machine/?comp=goalstate","http_user_agent":"WALinuxAgent/2.12.0.2","http_content_type":"text/xml","http_method":"GET","protocol":"HTTP/1.
1","status":200,"length":2078},"app_proto":"http","fileinfo":{"filename":"/machine/","gaps":false,"state":"CLOSED","sha256":"e27d21746235b93b84df005b95e854ef22453bc61d513ffd89e228bd37f77641","stored":false,"size":2078,"tx_id":…
log_line Sep 30 20:06:01 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:01.614320+0530","flow_id":364497316693777,"in_iface":"hn0","event_type":"http","src_ip":"172.29.0.18","src_port":55754,"dest_ip":"168.63.129.16","dest_
port":32526,"proto":"TCP","pkt_src":"wire/pcap","tx_id":0,"http":{"hostname":"168.63.129.16","http_port":32526,"url":"/vmSettings","http_user_agent":"WALinuxAgent/2.12.0.2","http_method":"GET","protocol":"HTTP/1.1","status":304,"len
gth":0,"request_headers":[{"name":"Accept-Encoding","value":"identity"},{"name":"Connection","value":"close"}],"response_headers":[{"name":"Server","value":"Microsoft-IIS/10.0"},{"name":"Date","value":"Tue, 30 Sep 2025 14:36:01 GM…
log_line Sep 30 20:06:01 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:01.607498+0530","flow_id":347815794165999,"in_iface":"hn0","event_type":"http","src_ip":"172.29.0.18","src_port":24308,"dest_ip":"168.63.129.16","dest_
port":80,"proto":"TCP","pkt_src":"wire/pcap","tx_id":0,"http":{"hostname":"168.63.129.16","url":"/machine/?comp=goalstate","http_user_agent":"WALinuxAgent/2.12.0.2","http_content_type":"text/xml","http_method":"GET","protocol":"HTT
P/1.1","status":200,"length":2078,"request_headers":[{"name":"Accept-Encoding","value":"identity"},{"name":"Connection","value":"close"}],"response_headers":[{"name":"Content-Type","value":"text/xml; charset=utf-8"},{"name":"Serve…
log_line Sep 30 20:06:01 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:01.617025+0530","flow_id":392056356935341,"in_iface":"hn0","event_type":"http","src_ip":"172.29.0.18","src_port":28248,"dest_ip":"168.63.129.16","dest_
port":32526,"proto":"TCP","pkt_src":"wire/pcap","tx_id":0,"http":{"hostname":"168.63.129.16","http_port":32526,"url":"/status","http_user_agent":"WALinuxAgent/2.12.0.2","http_method":"PUT","protocol":"HTTP/1.1","status":200,"lengt
h":0,"request_headers":[{"name":"Accept-Encoding","value":"identity"},{"name":"Content-Length","value":"568"},{"name":"Content-type","value":"application/json"},{"name":"Connection","value":"close"}],"response_headers":[{"name":"Se…
log_line Sep 30 20:06:01 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:01.618794+0530","flow_id":400525324334173,"in_iface":"hn0","event_type":"http","src_ip":"172.29.0.18","src_port":52670,"dest_ip":"168.63.129.16","dest_
port":32526,"proto":"TCP","pkt_src":"wire/pcap","tx_id":0,"http":{"hostname":"168.63.129.16","http_port":32526,"url":"/status","http_user_agent":"WALinuxAgent/2.12.0.2","http_method":"PUT","protocol":"HTTP/1.1","status":200,"lengt
h":0,"request_headers":[{"name":"Accept-Encoding","value":"identity"},{"name":"Content-Length","value":"1957"},{"name":"Content-type","value":"application/json"},{"name":"Connection","value":"close"}],"response_headers":[{"name":"S…
log_line Sep 30 20:06:07 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:07.663813+0530","flow_id":1995186146477417,"in_iface":"hn0","event_type":"fileinfo","src_ip":"168.63.129.16","src_port":80,"dest_ip":"172.29.0.18","des
t_port":16718,"proto":"TCP","pkt_src":"wire/pcap","http":{"hostname":"168.63.129.16","url":"/machine/?comp=goalstate","http_user_agent":"WALinuxAgent/2.12.0.2","http_content_type":"text/xml","http_method":"GET","protocol":"HTTP/1.
1","status":200,"length":2078},"app_proto":"http","fileinfo":{"filename":"/machine/","gaps":false,"state":"CLOSED","sha256":"e27d21746235b93b84df005b95e854ef22453bc61d513ffd89e228bd37f77641","stored":false,"size":2078,"tx_id":…
log_line Sep 30 20:06:01 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:01.618794+0530","flow_id":400525324334173,"in_iface":"hn0","event_type":"fileinfo","src_ip":"172.29.0.18","src_port":52670,"dest_ip":"168.63.129.16","d
est_port":32526,"proto":"TCP","pkt_src":"wire/pcap","http":{"hostname":"168.63.129.16","http_port":32526,"url":"/status","http_user_agent":"WALinuxAgent/2.12.0.2","http_method":"PUT","protocol":"HTTP/1.1","status":200,"length":0},"a
pp_proto":"http","fileinfo":{"filename":"/status","gaps":false,"state":"CLOSED","sha256":"f6774ef4c77fe78b47a02adb82de493c103ec462c2fbf36339ecdd20fff3ebd8","stored":false,"size":1957,"tx_id":0}} timestamp Sep 30, 2025 @ 20:13:54.40…
log_line Sep 30 20:06:01 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:01.617025+0530","flow_id":392056356935341,"in_iface":"hn0","event_type":"fileinfo","src_ip":"172.29.0.18","src_port":28248,"dest_ip":"168.63.129.16","d
est_port":32526,"proto":"TCP","pkt_src":"wire/pcap","http":{"hostname":"168.63.129.16","http_port":32526,"url":"/status","http_user_agent":"WALinuxAgent/2.12.0.2","http_method":"PUT","protocol":"HTTP/1.1","status":200,"length":0},"a
pp_proto":"http","fileinfo":{"filename":"/status","gaps":false,"state":"CLOSED","sha256":"c0a02efb7806becd3876edea258049e9bf3f30ea79b9ce8b284ba25ceea80f0b","stored":false,"size":568,"tx_id":0}} timestamp Sep 30, 2025 @ 20:13:52.71…
log_line Sep 30 20:06:07 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:07.682761+0530","flow_id":2042368193755106,"in_iface":"hn0","event_type":"fileinfo","src_ip":"172.29.0.18","src_port":29756,"dest_ip":"168.63.129.1
6","dest_port":32526,"proto":"TCP","pkt_src":"wire/pcap","http":{"hostname":"168.63.129.16","http_port":32526,"url":"/status","http_user_agent":"WALinuxAgent/2.12.0.2","http_method":"PUT","protocol":"HTTP/1.1","status":200,"length":
0},"app_proto":"http","fileinfo":{"filename":"/status","gaps":false,"state":"CLOSED","sha256":"5467ac658f322249fdccb01e0c457bbae41724ebdffbfd6bfbf58cac5526fd9a","stored":false,"size":1957,"tx_id":0}} timestamp Sep 30, 2025 @ 20:13:…
log_line Sep 30 20:06:07 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:07.671679+0530","flow_id":2035304252925988,"in_iface":"hn0","event_type":"fileinfo","src_ip":"172.29.0.18","src_port":33219,"dest_ip":"168.63.129.1
6","dest_port":32526,"proto":"TCP","pkt_src":"wire/pcap","http":{"hostname":"168.63.129.16","http_port":32526,"url":"/status","http_user_agent":"WALinuxAgent/2.12.0.2","http_method":"PUT","protocol":"HTTP/1.1","status":200,"length":
0},"app_proto":"http","fileinfo":{"filename":"/status","gaps":false,"state":"CLOSED","sha256":"847f6ec2df5bd1d332191ffddaf1ef63ed6aead7a7e7180c1414018f2cc811a2","stored":false,"size":568,"tx_id":0}} timestamp Sep 30, 2025 @ 20:13:5…
log_line Sep 30 20:06:00 pfsense newsyslog[15000]: logfile turned over due to size>500K timestamp Sep 30, 2025 @ 20:13:48.524 _id zKMUm5kB-YGlhpFFZ5zb _ignored - _index pfsense_logs _score 1
log_line Sep 30 20:06:20 pfsense suricata[45674]: {"timestamp":"2025-09-30T20:06:19.714985+0530","flow_id":1091033497740542,"in_iface":"hn0","event_type":"http","src_ip":"172.29.0.18","src_port":47611,"dest_ip":"168.63.129.16","dest
_port":80,"proto":"TCP","pkt_src":"wire/pcap","tx_id":0,"http":{"hostname":"168.63.129.16","url":"/machine/?comp=goalstate","http_user_agent":"WALinuxAgent/2.12.0.2","http_content_type":"text/xml","http_method":"GET","protocol":"HTT
P/1.1","status":200,"length":2078,"request_headers":[{"name":"Accept-Encoding","value":"identity"},{"name":"Connection","value":"close"}],"response_headers":[{"name":"Content-Type","value":"text/xml; charset=utf-8"},{"name":"Serve…
```

This ensures layered monitoring combining identity logs with network telemetry to strengthen security posture.

6.6 Troubleshooting & Fixes Documented

During implementation, several misconfigurations and errors were resolved, demonstrating robustness of the solution:

- **Null Client Secret** → Corrected Keycloak client configuration.
- **OIDC Unknown Provider** → Set idp_provider: oidc in Pomerium config.
- **x509 Validation Failure** → Imported Keycloak CA into system trust.
- **CSRF Token Errors** → Aligned redirect URIs and scopes.

Each issue was systematically diagnosed and resolved, resulting in a stable ZTNA authentication workflow.

6.7 Summary of Evidence

Validation Area	Evidence Type	Location / Output
Service startup	Systemd status checks	systemctl status keycloak, pomerium
Authentication flow	Browser test, successful login	https://app.aether.local
Log generation	JSON log files	/var/log/pomerium/*.json, /var/log/keycloak/*.json
SIEM integration	ELK dashboards & alerts	Authentication dashboard
Troubleshooting resolution	Fixed errors, documented changes	System logs & configuration history

The above evidence confirms that the SASE + ZTNA solution is **fully operational in the lab environment**, demonstrating secure SSO access, correct OIDC integration, strong TLS protection, centralized visibility, and network-level telemetry all achieved using open-source technologies.

7. Requirement ↔ Evidence Mapping Table

The table below establishes a clear **requirement-to-evidence traceability matrix**, mapping each requirement to the specific configuration, validation activity, and supporting proof gathered during the implementation and testing of the SASE + ZTNA solution. This structure ensures transparency, accountability, and ease of audit for security and compliance reviews.

Requirement	**Solution Implementation**	**Evidence Collected**	**Location / Reference**
1. Centralized identity-based authentication	Configured Keycloak (OIDC) as the IdP and integrated with Pomerium for authentication via Authorization Code Flow with PKCE	- Keycloak realm and client configuration - Successful SSO login via browser flow	- Keycloak Admin Console - /var/log/keycloak/keycloak.json
2. Secure TLS communication for all endpoints	Implemented TLS on Pomerium (app & authenticate hostnames) and Keycloak; trusted CA manually imported into IAM system	- TLS certificates in /etc/pomerium/certs/ and /etc/keycloak/certs/ - Successful HTTPS connections and trust validation	- System trust store (/usr/local/share/ca-certificates) - Browser test & OIDC flow
3. Zero Trust Network Access policy enforcement	Defined route policy in config.yaml restricting access to contractor1@aether.local; enforced identity header forwarding	- Policy block for non-authorized users - Header injection to upstream app	- /home/identity/config.yaml - Pomerium logs: /var/log/pomerium/pomerium.json
4. Single Sign-On (SSO) user experience	Enabled full OIDC redirection flow between Pomerium and Keycloak with session persistence	- Successful login flow test with redirect and callback handling	- Browser access to https://app.aether.local - Journalctl logs
5. Authentication logging	Enabled JSON logging in Keycloak & Pomerium;	- Log entries for sign-ins, policy decisions, and	- /var/log/keycloak/*.json - /var/log/pomerium/*.json

for SIEM ingestion	configured file paths and logrotate	OIDC flows - Filebeat/Fluent Bit-ready format	
6. Centralized visibility through ELK dashboards	Forwarded all JSON logs to ELK for correlation and detection engineering	- Authentication dashboards - Detection rule for Pomerium sign-in	- ELK UI (Authentication Dashboard) - Detection Rule Config
7. Network traffic monitoring for layered defense	Deployed pfSense with Suricata on the monitored VNet	- IDS/IPS logs showing TLS handshakes and access flows - Network telemetry indexed in ELK	- Suricata logs in ELK
8. Resilience through troubleshooting and error handling	Systematically resolved configuration issues (client secret, CA trust, CSRF tokens, Envoy conflicts)	- Documented error logs and fixes - Final stable service state	- System logs - Implementation notes

Key Takeaways

- **Every requirement is matched with concrete, verifiable evidence.**
- The mapping demonstrates coverage across **identity**, **encryption**, **access control**, **observability**, and **network security** all core pillars of the SASE + ZTNA architecture.
- All evidence references are traceable to configuration files, log locations, or dashboards, ensuring auditability.

8. Conclusion (Achievements, Gaps, Roadmap)

Achievements

The deployment of the SASE + ZTNA solution successfully demonstrated the feasibility of implementing **identity-aware, encrypted, and monitored access** to internal applications using open-source technologies. Key achievements include:

- **End-to-end SSO Integration:** Seamless authentication flow between Pomerium and Keycloak using OIDC Authorization Code with PKCE, enabling centralized identity enforcement.
- **Policy-based Access Control:** Route-level access restrictions through Pomerium, with identity headers forwarded to the upstream application for context propagation.
- **Strong TLS Security:** All client–proxy and proxy–IdP communications are protected with TLS, with local CA trust configured to support self-signed certificates.
- **Centralized Visibility:** Keycloak and Pomerium emit structured JSON logs that are ingested into ELK, providing real-time authentication dashboards and enabling detection rule creation.
- **Layered Monitoring:** Deployment of pfSense with Suricata adds network-level visibility, complementing identity telemetry to strengthen overall situational awareness.
- **Operational Stability:** All major configuration and interoperability issues (OIDC provider errors, CA trust problems, redirect mismatches, Envoy conflicts) were systematically identified and resolved, resulting in a stable and reproducible ZTNA environment.

These achievements validate that the prototype aligns with the core Zero Trust and SASE objectives, including centralized identity, encrypted access, strong policy enforcement, and integrated observability.

Gaps Identified

While the current prototype meets its functional goals, several gaps were identified when evaluated against production-grade requirements:

- **Scalability:** The solution runs on a single IAM node with no load balancing or high-availability configurations.

- **Federated Identity:** Keycloak operates as a standalone IdP; integration with enterprise identity providers (e.g., Azure AD, Okta) is not yet implemented.
- **Policy Granularity:** Access control is applied at the route level for individual users; group-based, attribute-based, and context-aware policies are not configured.
- **Certificate Lifecycle Management:** Certificates are manually generated and trusted; no automated issuance or renewal is in place.
- **Monitoring Depth:** ELK dashboards and detection rules cover sign-in activity but do not yet include advanced correlation, anomaly detection, or automated incident response.
- **Limited Application Coverage:** Only a single demo application is protected; broader application onboarding and ZTNA expansion remain pending.

These gaps are not shortcomings of the prototype itself but deliberate scoping decisions made to prioritize core functionality within the available time and open-source constraints.

Roadmap

To transition from a lab prototype to a production-ready SASE + ZTNA deployment, the following roadmap is proposed:

Short-Term Enhancements

- Integrate enterprise identity providers with Keycloak for federated SSO and MFA.
- Automate TLS certificate management using enterprise PKI or ACME workflows.
- Expand ELK detection content to include failed logins, anomalous geolocation access, and privilege escalation attempts.
- Harden configurations and conduct targeted security testing.

Mid-Term Enhancements

- Deploy Pomerium and Keycloak in a high-availability architecture with load balancing and session persistence.
- Implement granular role-based and attribute-based access control policies.

- Extend ZTNA protection to additional internal applications and services.
- Develop operational runbooks for incident response and access management workflows.

Long-Term Enhancements

- Integrate with CASB/RBI solutions to provide full SASE coverage including secure internet access.
- Implement distributed enforcement points to support remote and branch users with low-latency access.
- Introduce advanced SIEM/SOAR integrations for automated detection and response pipelines.
- Align policy and monitoring configurations with regulatory compliance frameworks and enterprise SOC procedures.