Introduction to the Shared Signals Framework

Thomas Darimont

Shared Signals Working Group
OpenID Foundation

Thomas Darimont

- CTO | Identity Tailor GmbH
- Digital Identities ❤️ Standards
- Open Source Enthusiast
- Official Keycloak Maintainer
- OpenID Foundation Certification Team

@thomasdarimont
thomas@identity-tailor.de

Shared Signals Framework

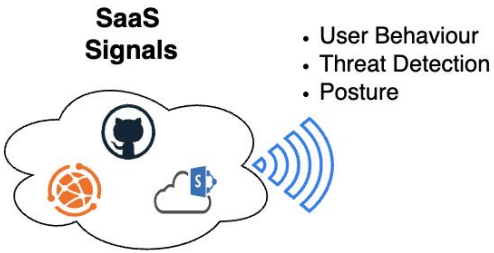
The **OpenID Shared Signals Framework (SSF)** provides a **standardized protocol** for identity systems to **communicate events** in a **trusted way** between parties to **improve security and user experience**.

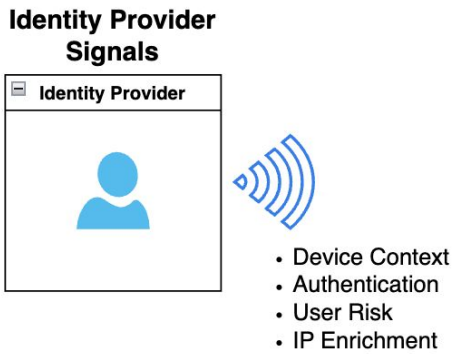
Shared Signals Working Group
OpenID Foundation

Goals

- Enable ***real-time*** communication of ***critical security events***
- *Reduce complexity* by *standardizing* ***event formats*** and ***delivery mechanisms***
- Enhance ***interoperability*** across ***identity systems***

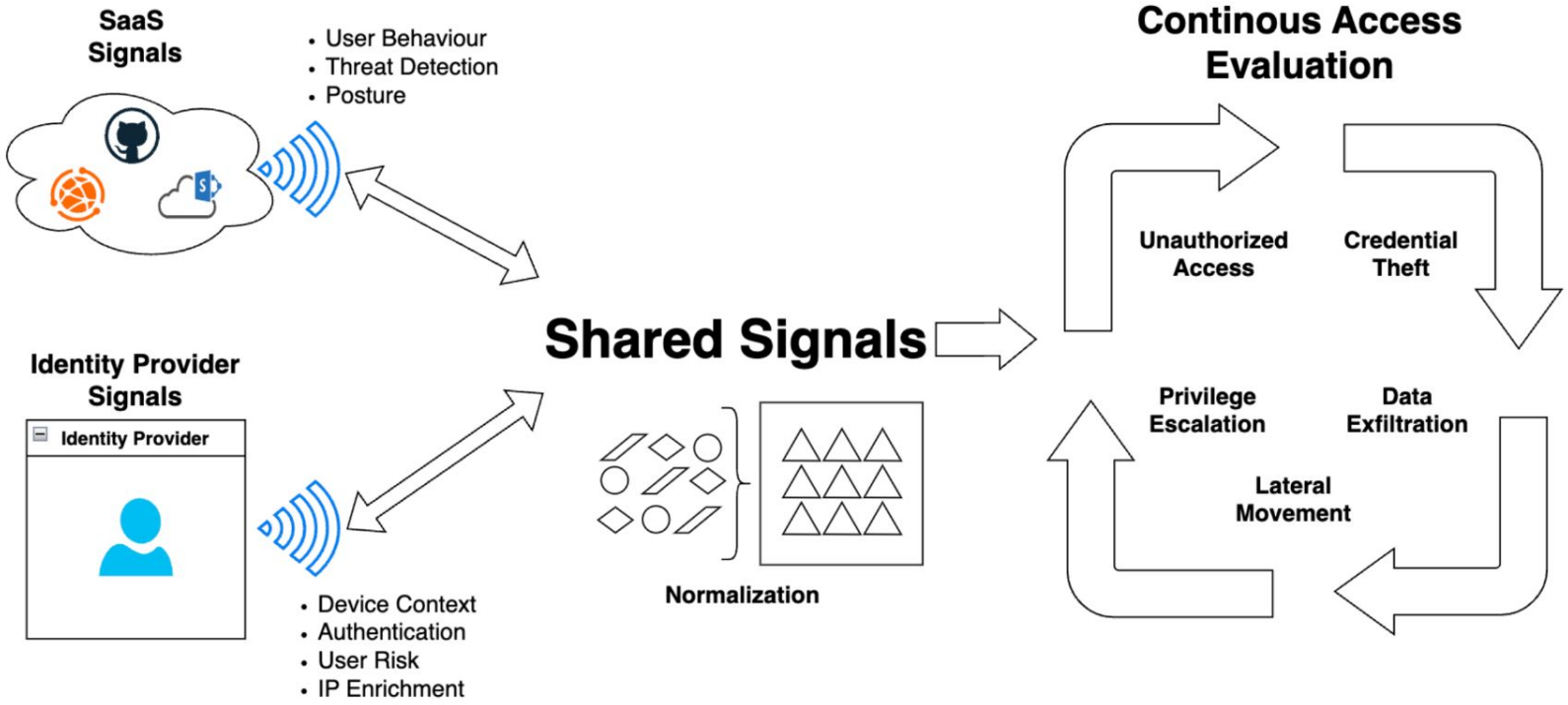
Enhancing SaaS Security





Credits to AppOmni / Okta

Enhancing SaaS Security



Use Case Examples

- **Real-Time Session Revocation**
  Revoke sessions instantly when risk conditions change, ensuring real-time access control.

- **Compromised Account Alert**
  Receive notifications when an IdP detects account compromise, triggering security measures.

- **Automated User Deprovisioning**
  Sync user lifecycle events to revoke access upon termination, preventing orphaned accounts.

Building Blocks

- **Security Event**

  Event resulting from observed behavior (user, system, device)

- **Transmitter**

  System emitting an *security event*, e.g. an Identity Provider

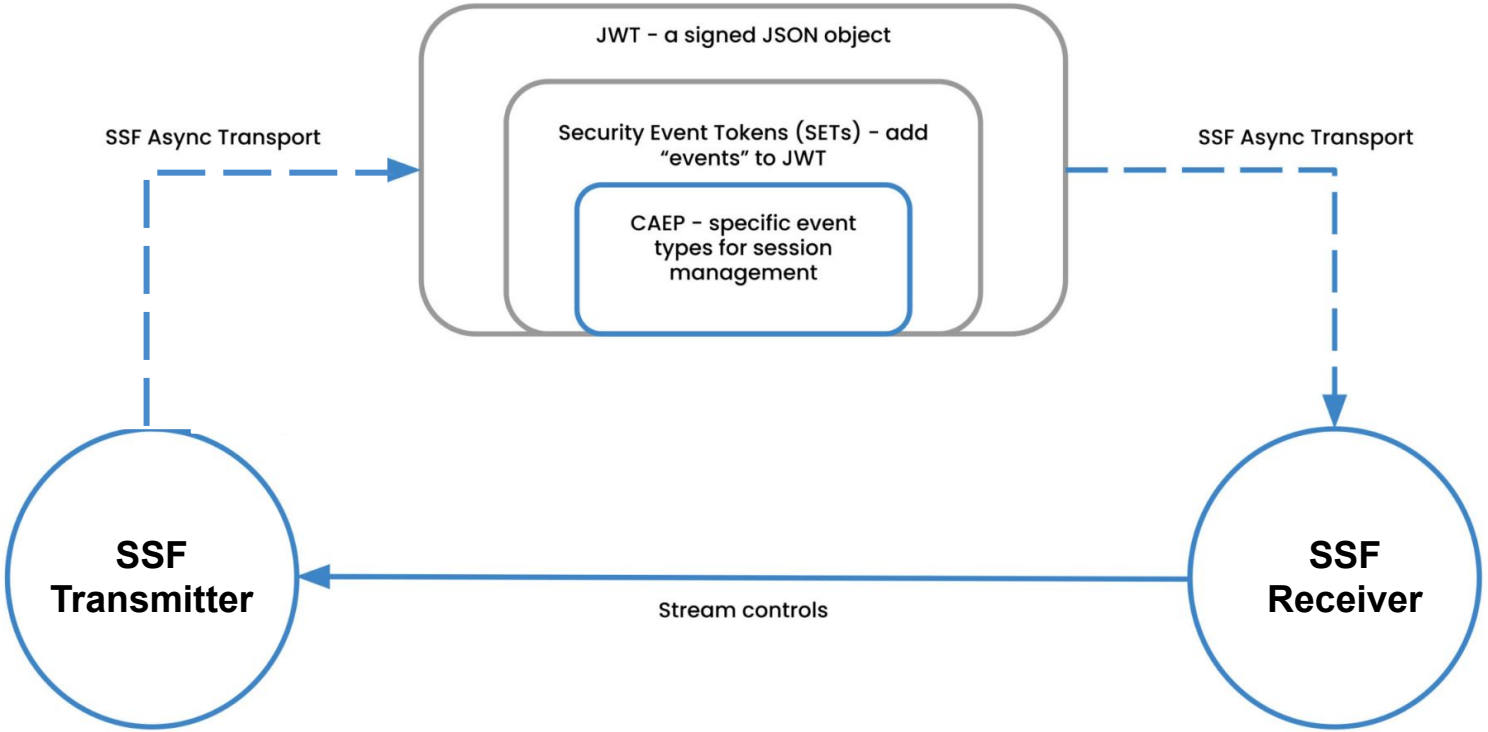
- **Receiver**

  System consuming the *security event*, e.g., a Service Provider

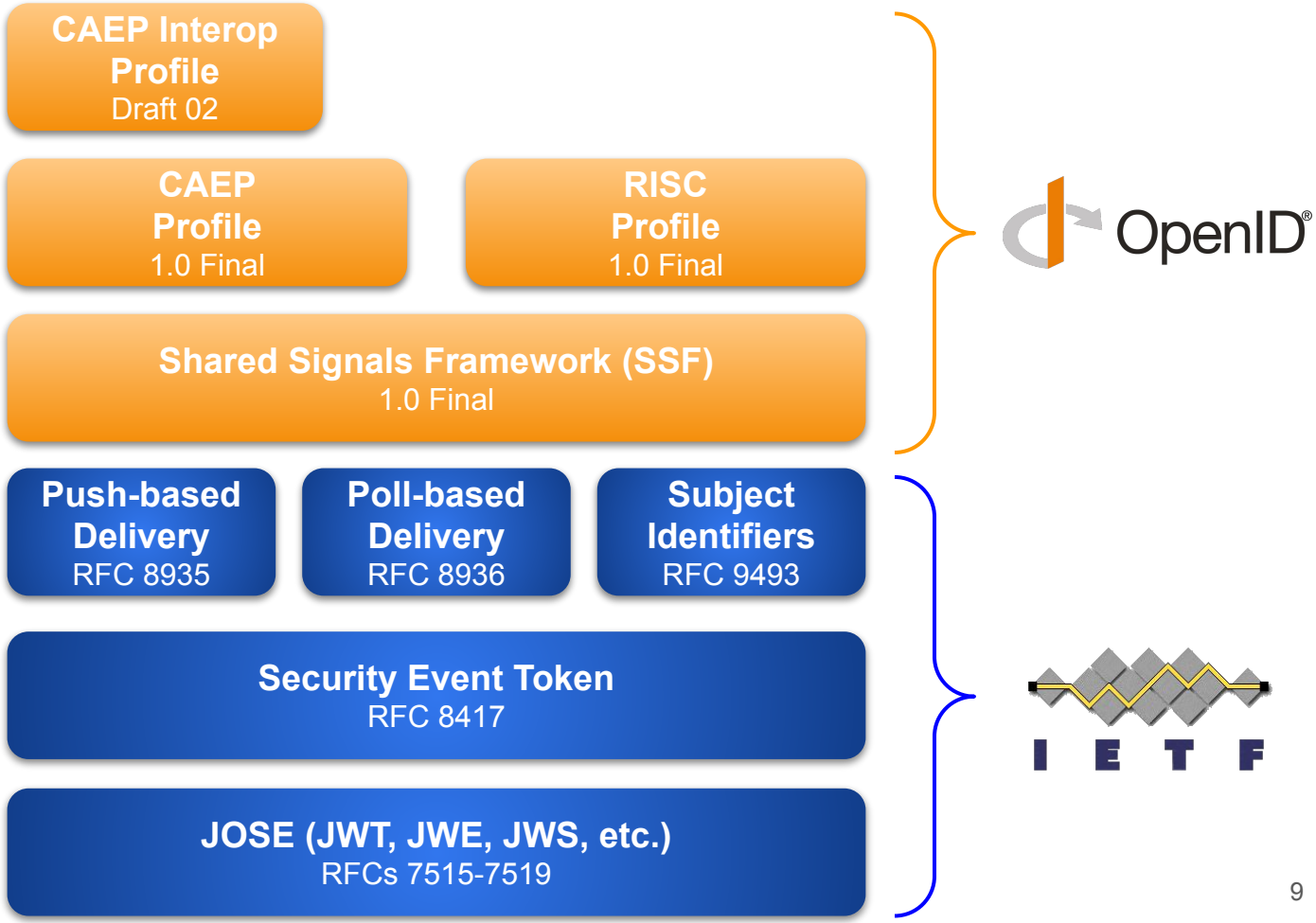
- **Stream**

  S*ubscription* for certain security events controlled by a *Receiver*, managed by the *Transmitter*

Topology


JWT - a signed JSON object
Security Event Tokens (SETs) - add "events" to JWT
CAEP - specific event types for session management
SSF Async Transport
SSF Async Transport
SSF Transmitter
SSF Receiver
Stream controls

SSF Protocol Stack



Security Event Token and Subject Format

Security Event Token (SET) RFC 8417

- A **JSON Web Token (JWT)** based **format** for conveying **security-related events**
- Includes event type, event specific payload, subject identifier, and metadata
- *Signed* by Transmitter for non-repudiation and integrity protection

Subject Identifiers RFC 9493

- A standardized way to reference **identities** across systems
- Identity can be a **person**, **device**, **group** or **organization**, tenant, session, etc.
- Many Subject Identifier **formats**: `email`, `phone_number`, `iss_sub`, `opaque`, etc.

Security Event Token (SET) Example


```
eyJhbGciOiJSUzI1NiIsImtpZCI6IjllMjJlMjc2LWQzYTQtNGE2OS1hZDA4LWQyNmNmNWI0Y2ExOSIsInR5cCI6InNlY2V2ZW50K2p3dCJ9.eyJpc3MiOiJodHRwczovL3NzZi5jYWVwLmRldi8iLCJhdWQiOiJodHRwczovL3NzZi1yZWNlaXZlci5rZXljbG9hay5vcmciLCJqdGkiOiJOams0TURRek4yTXRZekV6TnkwME5qQTFMVGszWWpRdE5ETmpOVFEyWldZd1ltRXoiLCJpYXQiOjE3Njk3NTk4NzIsInN1Yl9pZCI6eyJlbWFpbCI6InRlc3RlckBsb2NhbC50ZXN0IiwiZm9ybWF0IjoiZW1haWwifSwiZXZlbnRzIjp7Imh0dHBzOi8vc2NoZW1hcy5vcGVuaWQubmV0L3NlY2V2ZW50L2NhZXAvZXZlbnQtdHlwZS9zZXNzaW9uLXJldm9rZWQiOnsiZXZlbnRfdGltZXN0YW1wIjoxNzY5NzU5ODcyfX19.UEcDS9b31U3VJRhz5AlLXvrhWH_gyG0XlbVcQx1xvVK5gkagQEALfgIYs2WUExQQ6M2qxg6mN5LBAT27FxYg5lwf3xx91FaETKjD6UKUUNnD4Z9E43WFEkxr7-1Ib0Ff_ZoN-qIR6rAlaNyWXw0t6ZBEa_MRKSL_8WKT2cuftZas-y4n8pKX8BbPaDjlpyjTXN94aLA3cP32H_rW4imb_CaWOXD7-k_B5nkJPE7usEwowP-UL_lTDurlNutIleUEwLMvJ5Q3UuDqrIuOELLj3YGRblTQnH5RRoMVeCClZ1N1mrCs7Dd8rW6XGUrGOzKQsnC5NLLmAGPXiabNP2MvwA
```


Header

```
{
  "alg": "RS256",
  "kid": "9e22e276-d3a4-4a69-ad08-d26cf5b4ca19",
  "typ": "secevent+jwt"
}
```

SET Token Type

Payload

```
{
  "iss": "https://ssf.caep.dev/",
  "aud": "https://ssf-receiver.keycloak.org",
  "jti": "Njk4MDQzN2MtYzEzNy00NjA1LTk3YjQtNDNjNTQ2ZWYwYmEz",
  "iat": 1769759872,
  "sub_id": {
    "email": "tester@local.test",
    "format": "email"
  },
  "events": {
    "https://schemas.openid.net/secevent/caep/event-type/session-revoked": {
      "event_timestamp": 1769759872
    }
  }
}
```

SET Issuer

SET Audience

SET Subject

SET Event Type

Signature

f31XicNjPrHorTGB_Q_FOEJQ…..

SET Event Profiles*: CAEP, RISC

CAEP (Continuous Access Evaluation Profile) 1.0 Final

- **SSF profile** that enables **continuous monitoring** and **evaluation of access decisions**
- Example events: session-revoked, token-claims-changed, risk-level-changed

RISC (Risk and Incident Sharing and Coordination) 1.0 Final

- **SSF profile** focused on disaster mitigation and is related to **security risks and incidents**
- Example events: credential-compromised, credential-change-required, account-disabled, etc.

***) Event Profile = Set of Event Definitions**

CAEP & RISC Event Overview

CAEP

Continuous Access Evaluation Profile

- **Session Revoked**
- Token Claims Change
- Credential Change
- Assurance Level Change
- **Device Compliance Change**
- Session Established
- Session Presented
- **Risk Level Change**

RISC

Risk Incident Sharing and Coordination Profile

- Account Credential Change Required
- **Account Purged | Disabled | Enabled**
- Identifier Changed | Recycled
- **Credential Compromise**
- Opt In
- **Opt Out Initiated** | Cancelled | Effective
- Recovery Activated | Information Changed

SSF Transmitter Metadata

https://ssf.caep.dev/**.well-known/ssf-configuration**


```
{
   "issuer": "https://ssf.caep.dev/",
   "delivery_methods_supported": [
      "urn:ietf:rfc:8935",
      "urn:ietf:rfc:8936"
   ],
   "configuration_endpoint": "https://ssf.caep.dev/ssf/streams",
   "status_endpoint": "https://ssf.caep.dev/ssf/status",
   "jwks_uri": "https://ssf.caep.dev/.well-known/jwks.json",
   "spec_version": "1_0-ID2",
   "authorization_schemes": [
      {
         "spec_urn": "urn:ietf:rfc:6749"
      }
   ],
   "verification_endpoint": "https://ssf.caep.dev/ssf/verify"
}
```


Metadata Endpoint

Issuer

Delivery Method

Stream Management

Stream Control

Keys

Stream Verification

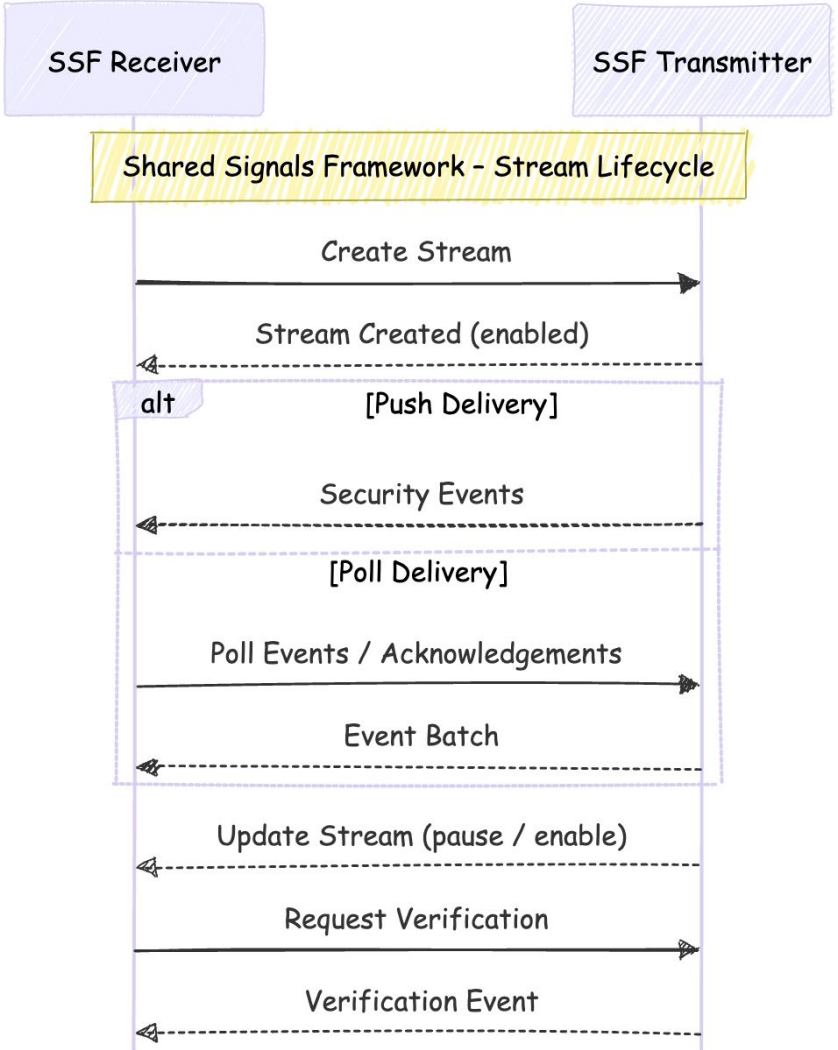
SET Delivery Methods

- **Push Delivery Method**
  - Push-Based Security Event Token (SET) Delivery Using HTTP **RFC 8935**
  - Receivers provides **Push-Endpoint** to Transmitter in stream definition
  - Transmitter sends events to **Push-Endpoint** via HTTP POST
  - Enables **Real-Time Event Delivery**

- **Poll Delivery Method**
  - Poll-Based Security Event Token (SET) Delivery Using HTTP **RFC 8936**
  - Transmitter provides **Poll-Endpoint**
  - Receiver periodically retrieves events from **Poll-Endpoint** via HTTP POST
  - Enables **Async Event Delivery & Batching**

Stream Lifecycle

- **Stream Management**
  Managed via Stream Configuration Endpoint
- **Stream Verification**
  Check Stream communication
- **Stream Status Update**
  Status Managed via Transmitter or Receiver



Create a Stream with Push Delivery

```
POST {{transmitterIssuer}}/ssf/streams
Authorization: Bearer {{transmitterToken}}
Content-Type: application/json
```

Auth Token

Stream Configuration

```
{
  "description": "This field is optional. Remove this field if not needed.",
  "delivery": {
    "method": "urn:ietf:rfc:8935",
    "endpoint_url": "https://id.acme.dev/realms/myrealm/ssf/push/caepdev",
    "authorization_header": "dummyAuthToken"
  },
  "events_requested": [
    "https://schemas.openid.net/secevent/caep/event-type/session-revoked",
    "https://schemas.openid.net/secevent/caep/event-type/credential-change",
    "https://schemas.openid.net/secevent/caep/event-type/device-compliance-change",
    "https://schemas.openid.net/secevent/caep/event-type/token-claims-change",
    "https://schemas.openid.net/secevent/caep/event-type/assurance-level-change"
  ]
}
```

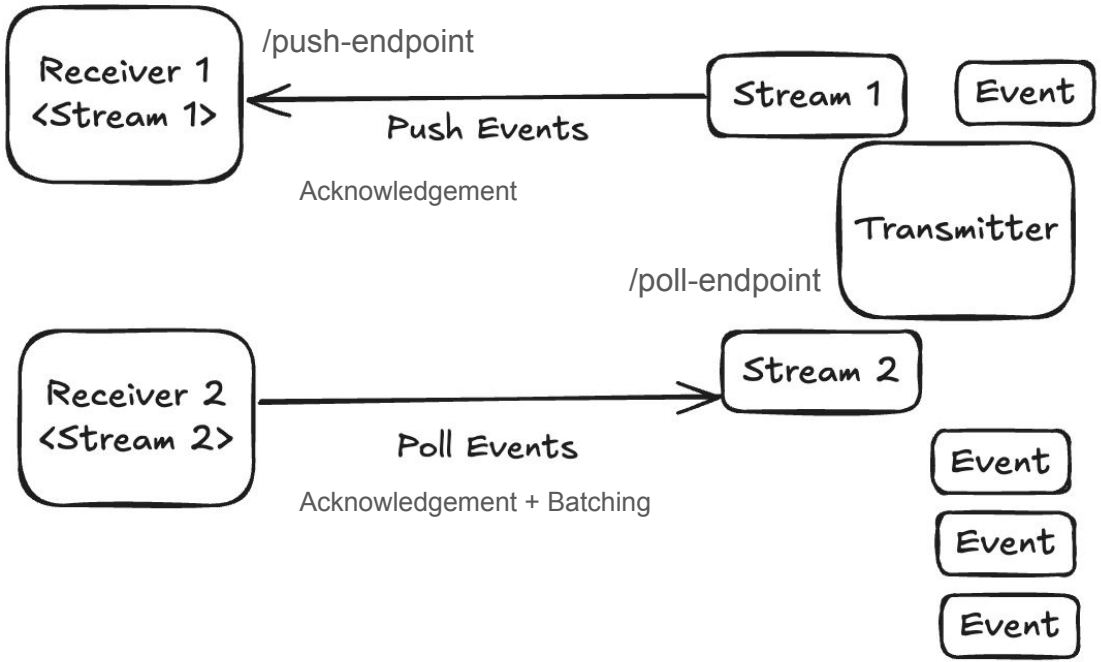
Delivery Method

Requested Events

SSF Event Delivery

Delivery via Push

Delivery via Polling



Example SET Delivery via Push

```
POST {{issuer}}/ssf/push/{{receiverAlias}}
Authorization: dummyAuthToken
Accept: application/json
Content-Type: application/secevent+jwt

eyJhbGci...encoded SET....
```

Push Endpoint on Receiver

Push Auth Token

SET Content Type

Encoded SET JWT

Example SET Delivery via Poll

```
POST {{transmitterIssuer}}/ssf/streams/poll
Authorization: Bearer {{transmitterToken}}
Content-Type: application/json
Accept: application/json

{"returnImmediately":true,"maxEvents":100}
```

Poll Endpoint on Transmitter

Transmitter Auth Token

Poll Request

Shared Signals Framework Adopters (*)

- AppOmni
- Cisco
- Delinea
- Google
- IBM
- Jamf
- Okta **
- Omnissa
- SailPoint
- Saviynt
- SGNL / CrowdStrike
- Thales
- WinMagic
- Disney
- Apple **

*) based on Participation survey at Gartner IAM SSF Interop Event January 2025 or **) Product Documentation

Shared Signals Framework Overview

Risk Incident Sharing and Coordination Profile (RISC)

Account Security Events

- Account disabled
- Account suspended
- Credentials Compromised
- …

Continuous Access Evaluation Profile (CEAP)

Session Management Events

- Session Revoked
- Token Claims Changed
- Risk Level Changed
- …

SCIM Events*

Entity Provisioning Events

- Account Created
- Account Updated
- Account Deleted
- …

Shared Signals Framework

- Asynchronous Publish/Subscribe Webhook Framework
- Stream of Security Event Tokens (SETs) - JWT format
- Subject identification (e.g. User, Group, Org, Tenant)
- Event Stream Management
- Push & Poll Delivery/Transport with Acknowledgement

Credits Sailpoint

*) SCIM Profile for Security Event Tokens IETF Draft

Shared Signals Framework

&

Keycloak: SSF Use Cases

- **Keycloak as SSF Transmitter**
  - Keycloak could emit CAEP/RISC events to interested parties (IdPs, SaaS apps)
  - Keycloak could notify “child” Identity providers about account changes, suspension
  - Challenges ****
    i. Simple Stream Management and Optional Scalable Event Storage
    ii. Polling / **Push** Infrastructure

- **Keycloak as SSF Receiver**
  - Keycloak can receive/fetch CAEP/RISC/SCIM events from SSF Transmitters
  - Identity Providers could propagate Session revocations, Account removal/suspension
  - Challenges ***
    i. SSF Stream Management Client
    ii. Event Ingestion and Event-Handling Infrastructure

caep.dev

Transmitter Receiver Learn Developer NEW About Register

Brought to you by sgnl

caep.dev is a service that enables Shared Signals Framework developers to test their Transmitters and Receivers. It implements the SSF draft specification and the CAEP draft specification required to operate the Transmitter and Receiver.

Register for a caep.dev access token

Start Transmitting

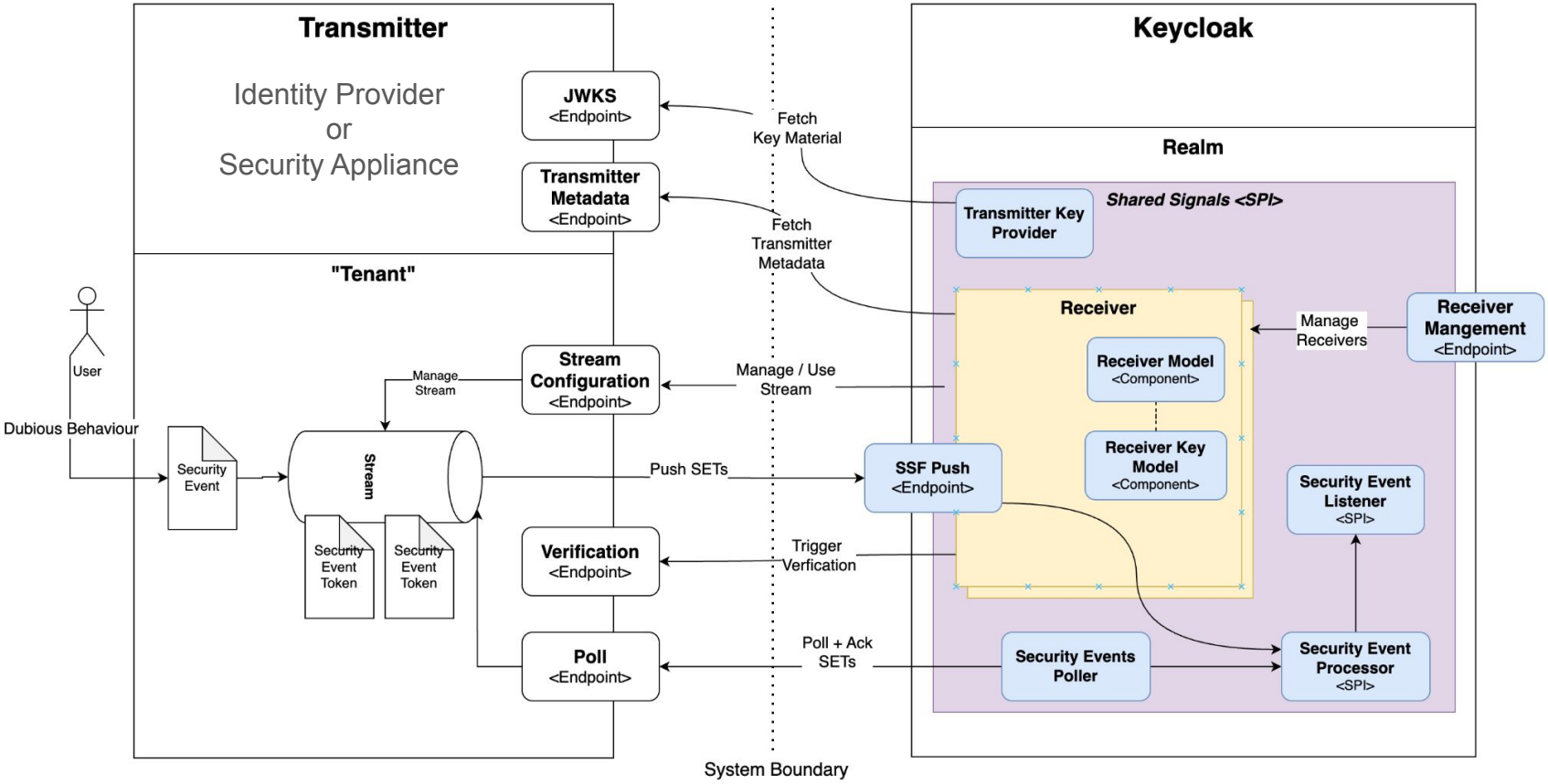
Use the Transmitter to generate and send CAEP events to your Receiver

Start Receiving

Use the Receiver to collect events from your Transmitters

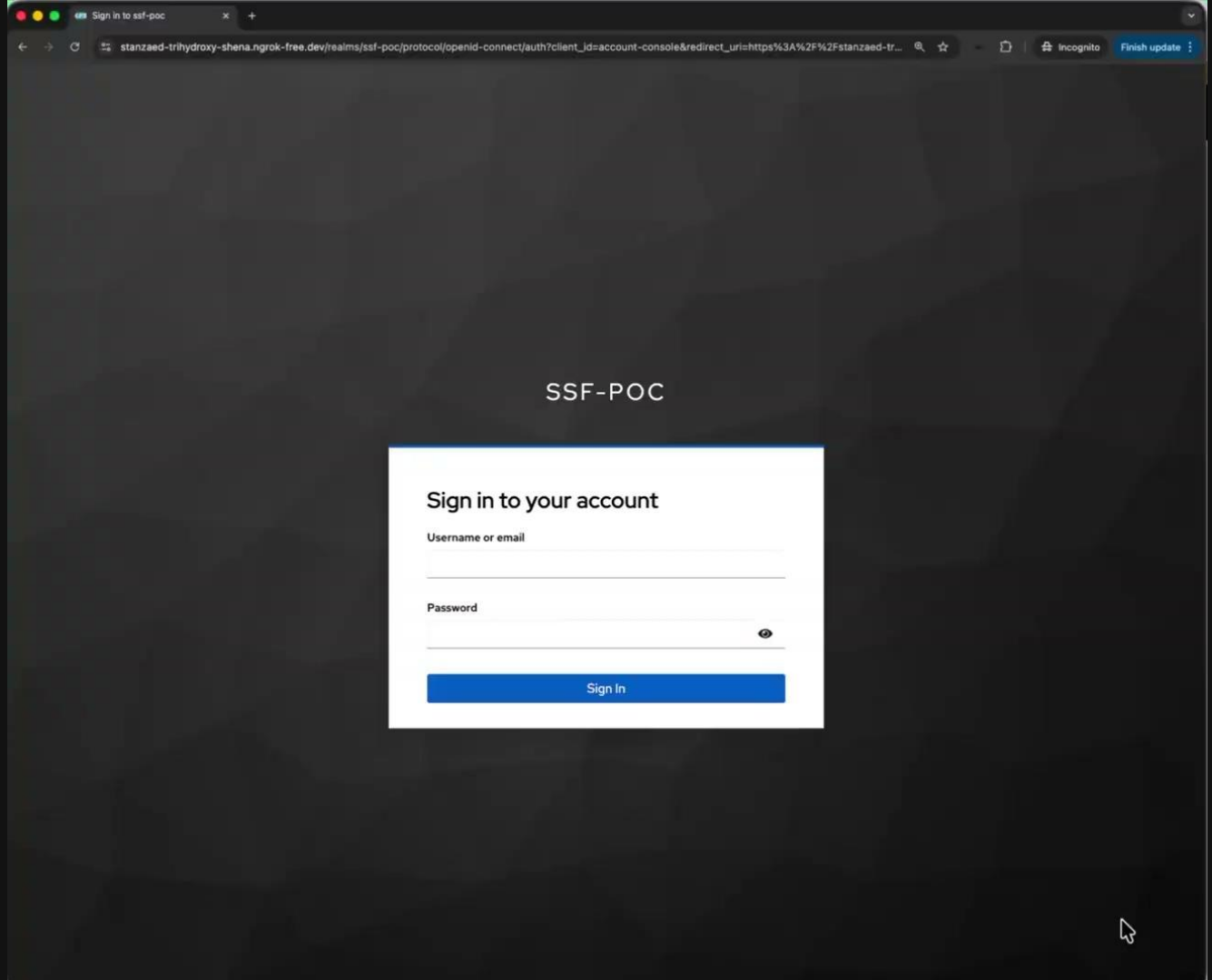
New: Join the discussion and track or report issues on GitHub!

Keycloak as SSF Receiver



Keycloak Shared Signals Framework Support

PR with PoC: https://github.com/keycloak/keycloak/pull/43950


Sign in to ssf-poc
stanzaed-trihydroxy-shena.ngrok-free.dev/realms/ssf-poc/protocol/openid-connect/auth?client_id=account-console&redirect_uri=https%3A%2F%2Fstanzaed-tr...
Incognito
Finish update
SSF-POC
Sign in to your account
Username or email
Password
Sign In

Shared Signals Framework Summary

- SSF standardizes real-time event delivery for identity systems
- Flexible and robust event sharing as SET via CAEP, and RISC
- Secure Push and Poll Delivery Mechanisms
- *Continuous Access Evaluation* and *Compromised Account Mitigation*
- SSF Receiver Support coming to Keycloak soon

Please Join!  **Shared Signals Working Group**
OpenID Foundation

Thank you!

Questions?

Feedback?

❤️

OpenID®

Contact thomas.darimont@oidf.org