

An Introduction to the Shared Signals Framework

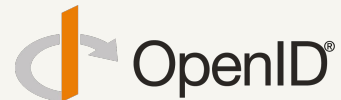
Real-time security events between identity providers and relying parties — and how Keycloak speaks them

Thomas Darimont Managing Director, Identity Tailor GmbH · Keycloak maintainer

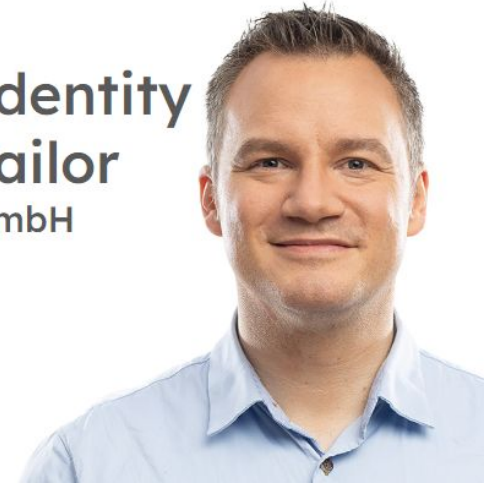
ABOUT ME

Thomas Darimont

- Managing Director, Identity Tailor GmbH
- Official Keycloak maintainer
- OpenID Foundation certification team
- Open source and open standards enthusiast
- Java User Group Saarland organizer



**Identity
Tailor**
GmbH



01 · WHY IT MATTERS

You revoke a user's session in Keycloak.

When does the app notice?

Usually at the next token refresh — minutes, hours, or never.

Downstream apps are the last to know

Account disabled

The offboarded employee keeps working in SaaS apps until their session there ends.

Credential compromised

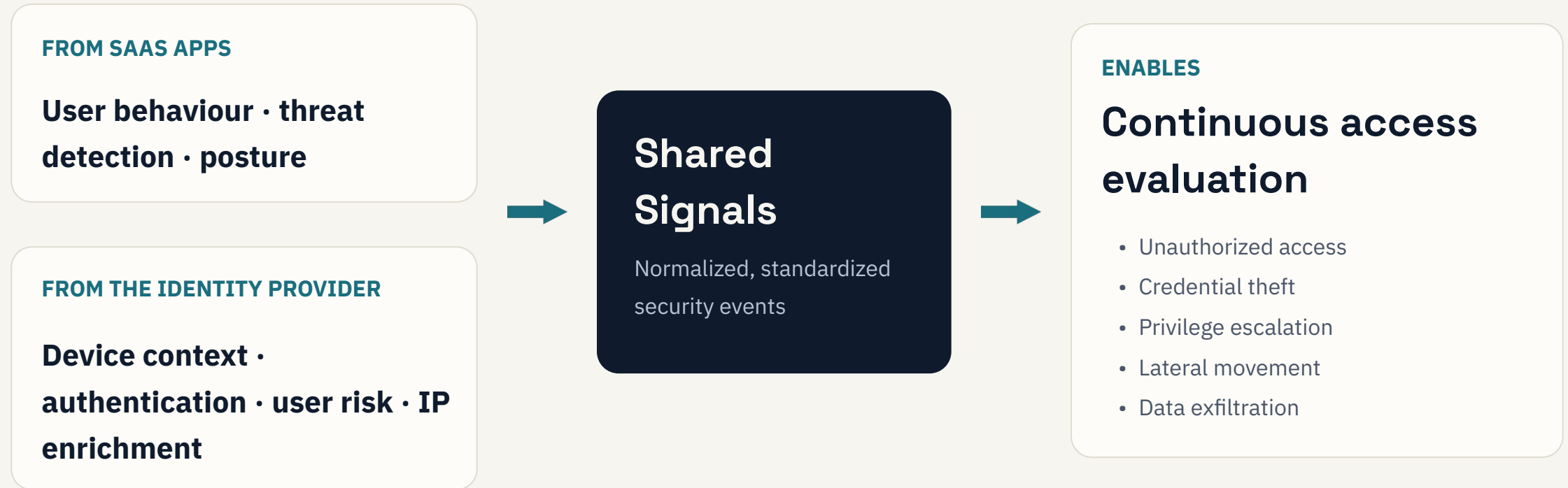
A password reset doesn't end the sessions already open downstream.

Device non-compliant

Posture changes, but apps keep trusting the login from before.

Today's workarounds: front-/backchannel logout, short token lifetimes, per-vendor webhooks, bespoke polling endpoints, a Kafka topic per integration.

Signals flow in both directions



The OpenID Shared Signals Framework is a standard protocol for identity systems

- to exchange security events
- in real-time,
- in a trusted way.

REAL-TIME

Critical security events delivered as they happen

STANDARDIZED

One event format and delivery model for every vendor

INTEROPERABLE

Across identity providers, SaaS apps and security tools

Three things SSF enables

01

Real-time session revocation

End sessions everywhere the moment risk changes.

CAEP session-revoked

02

Compromised account alerts

Hear about it as soon as an identity provider detects a takeover.

RISC credential-compromise

03

Automated deprovisioning

Revoke access at termination
→ no orphaned accounts.

RISC account-disabled

Four building blocks



Security Event

Something observed about a user, device or session



Transmitter

Emits security events, e.g. an identity provider



Receiver

Consumes security events — e.g. a SaaS app



Stream

A receiver's subscription to selected events, run by the transmitter

How the pieces connect



Built on proven standards

OPENID FOUNDATION	CAEP Interop Profile			1.0 Draft
	CAEP	1.0 Final	RISC	1.0 Final
	SCIM events			RFC 9967
	Shared Signals Framework (SSF)			1.0 Final
IETF	Push delivery	RFC 8935	Poll delivery	RFC 8936
	Subject IDs			RFC 9493
	Security Event Token (SET)			RFC 8417
	JOSE — JWT, JWS, JWE			RFC 7515–7519

Events travel as signed tokens

RFC 8417

Security Event Token

A JWT that carries a security event: its type, payload, subject and metadata.

Signed by the transmitter for integrity and non-repudiation.

RFC 9493

Subject identifiers

A standard way to name who or what an event is about: a person, device, group, organization, tenant, session or IP address.

email

phone_number

iss_sub

ip

opaque

...

Anatomy of a SET

```
// JOSE header
{ "alg": "RS256", "kid": "9e22e276...", "typ": "secevent+jwt" }
// payload
{
  "iss": "https://ssf.caep.dev",
  "aud": "https://ssf-receiver.keycloak.org",
  "jti": "Njk4MDQzN2MtYzEzNy00NjA1...",
  "iat": 1769759872,
  "sub_id": { "format": "email", "email": "tester@local.test" },
  "events": {
    "https://schemas.openid.net/secevent/caep/event-type/session-revoked": {
      "event_timestamp": 1769759872
    }
  }
}

// + signature with the transmitter's key
```

typ

Marks the JWT as a Security Event Token

iss • aud

Who sent it, and which receiver it's for

sub_id

The subject, as an RFC 9493 identifier

events

Event type URI → event-specific payload

Event profiles give signals meaning

OPENID • 1.0 FINAL

CAEP

Continuous Access Evaluation Profile

Focus: Sessions and Access context

- Session revoked
- Token claims change
- Credential change
- Assurance level change
- Device compliance change
- Session established
- Session presented
- Risk level change

OPENID • 1.0 FINAL

RISC

Risk Incident Sharing and Coordination

Focus: Account-level Incidents

- Credential compromise
- Account credential change required
- Account purged / disabled / enabled
- Identifier changed / recycled
- Opt in / opt out
- Recovery activated / info changed

IETF • RFC 9967

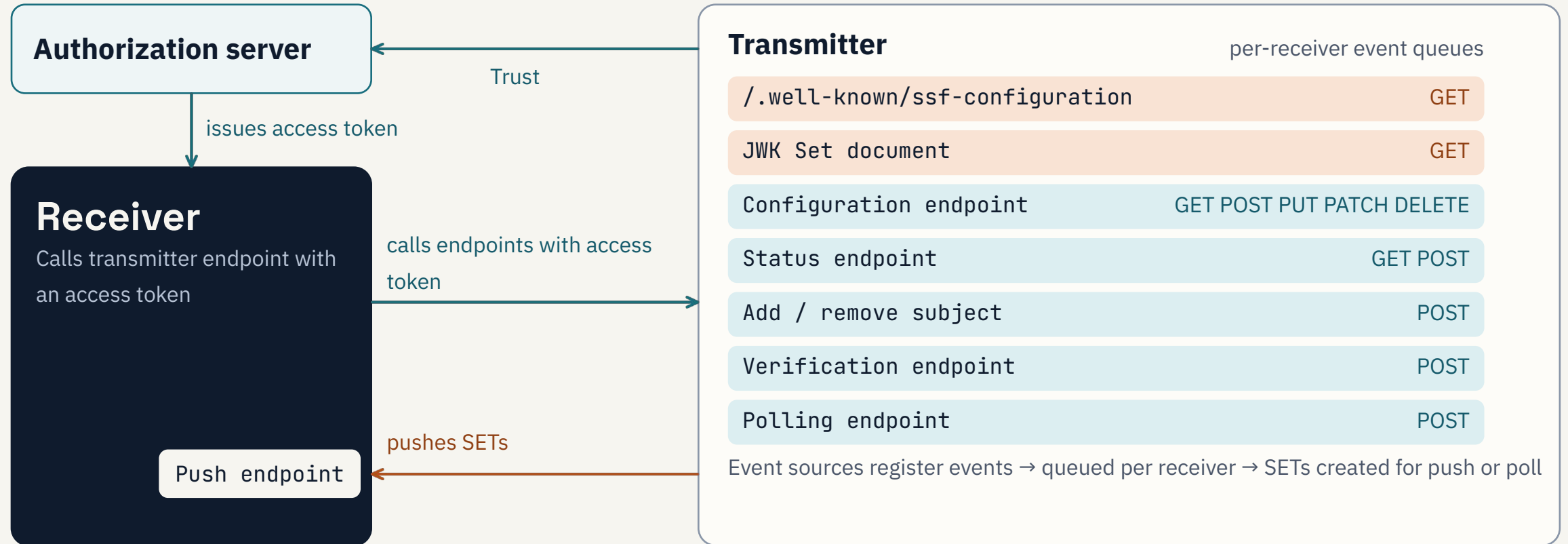
SCIM Events

SCIM profile for Security Event Tokens

Focus: Provisioning Lifecycle

- Account created
- Account updated
- Account deleted
- ...

Anatomy of a transmitter



Discovery: transmitter metadata

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

delivery_methods_supported

Push (RFC 8935) and/or poll (RFC 8936)

configuration_endpoint

Create, read, update, delete streams

status_endpoint

Read or change a stream's status

verification_endpoint

Request a verification event

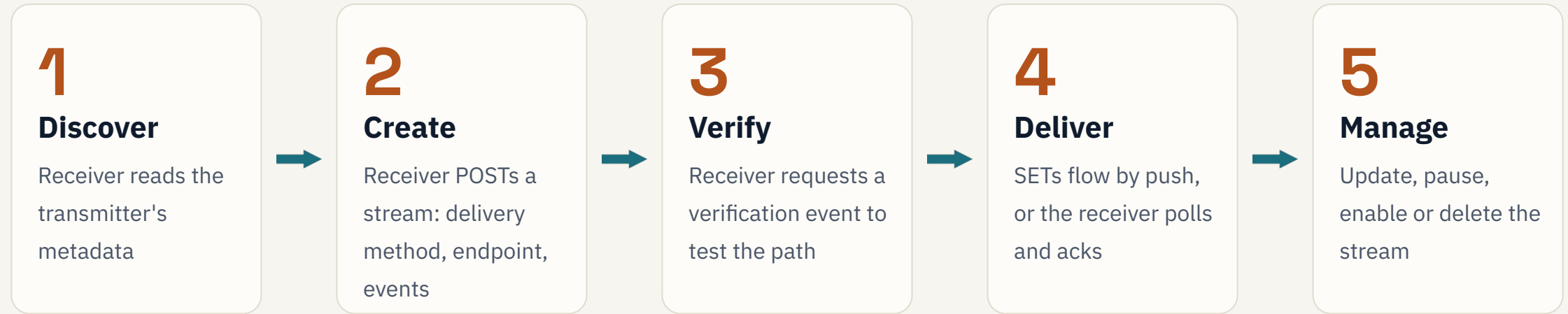
jwks_uri

Keys to verify SET signatures

authorization_schemes

OAuth 2.0 bearer tokens

A stream's lifecycle



Receiver-driven: streams are managed by the receiver through the transmitter's configuration and status endpoints, secured with OAuth 2.0 bearer tokens.

Transmitter-driven: streams are managed by the transmitter's configuration and status endpoints, secured with OAuth 2.0 bearer tokens.

Creating a push stream

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

{
  "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"
  ]
}
```

Authorization

OAuth 2.0 access token for the transmitter

delivery

Delivery mechanism

method

rfc:8935 = push, rfc:8936 = poll

endpoint_url

Where the transmitter pushes SETs

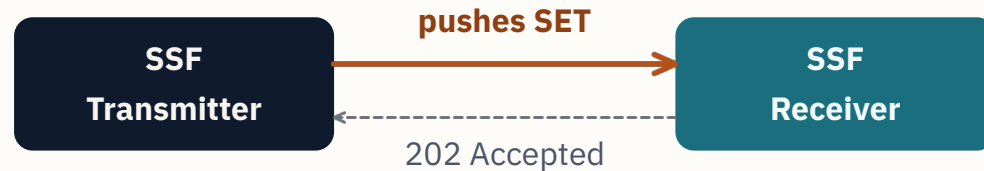
events_requested

What the receiver wants; the reply lists events_delivered

Two ways to deliver events

Push

RFC 8935



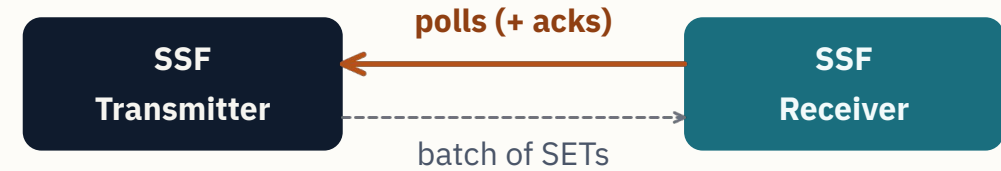
```
POST https://receiver.example/ssf/push
Authorization: {pushAuthHeader}
Content-Type: application/secevent+jwt

eyJhbGciOiJIUzUzI1NiIs...
```

Authorization header is optional; the receiver acknowledges with HTTP 202.

Poll

RFC 8936



```
POST {transmitter}/ssf/streams/poll
Authorization: Bearer {token}
Content-Type: application/json

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

Batching · return now or wait · fetch and acknowledge · acknowledge only · report errors

Who's adopting SSF

AppOmni	Apple **	Cisco	Delinea
Disney	Google **	IBM	Jamf
Keycloak	Okta **	OmniSSA	SailPoint
Saviynt	SGNL / CrowdStrike	Thales	WinMagic
Authentik	Zscaler**	cloudflare**	Palo Alto Networks**

Shared Signals in Keycloak

A realm as SSF transmitter — and as receiver next



Two roles for Keycloak

Shipped with 26.7.0 • experimental

Keycloak as transmitter

- Emits CAEP and RISC events to SaaS apps and other identity providers
- Notifies downstream “child” IdPs about revoked sessions, credential updates, account changes and suspensions

Challenges: simple stream management, scalable event storage, push and poll infrastructure

Proof of concept • tentative

Keycloak as receiver

- Receives CAEP, RISC and SCIM events from external transmitters
- Lets upstream IdPs propagate session revocation, account removal and suspension

Challenges: a stream management client, event ingestion and handling

Test against caep.dev

The screenshot shows the caep.dev website with a dark blue background. At the top, the logo 'caep.dev' is on the left, and navigation links 'Transmitter', 'Receiver', 'Learn', 'Developer' (with a 'NEW' badge), 'About', and 'Register' are in the center. On the right, there is a refresh icon and the text 'Brought to you by' followed by the 'sgnl' logo. The main content area features a central circular graphic with a white 'i' and concentric circles. Above it, text explains that caep.dev is a service for Shared Signals Framework developers to test transmitters and receivers, implementing the SSF draft specification and the CAEP draft specification. Below the central graphic, there are two main buttons: 'Start Transmitting' (cyan) on the left and 'Start Receiving' (orange) on the right. Each button has a sub-description: 'Use the Transmitter to generate and send CAEP events to your Receiver' and 'Use the Receiver to collect events from your Transmitters' respectively. A link 'Register for a caep.dev access token' is positioned above the central graphic. At the bottom, a 'New' announcement invites users to join a discussion and track or report issues on GitHub.

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!

Hosted SSF transmitter and receiver for developers

Send CAEP events to your receiver

Collect events from your transmitter

DEMO 1 OF 4

Keycloak as SSF transmitter

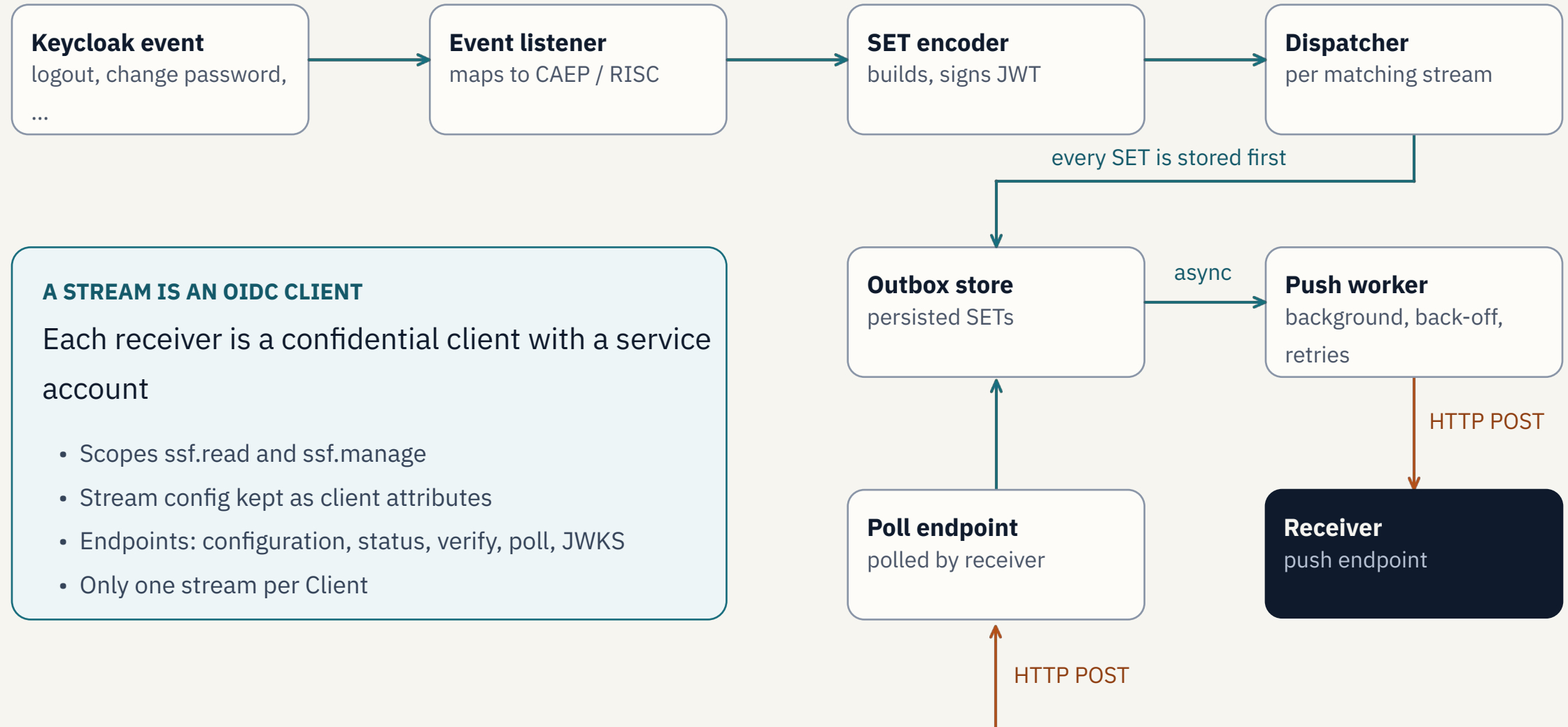
TRY IT YOURSELF

```
--features=preview,ssf · quay.io/keycloak/keycloak:nightly
```

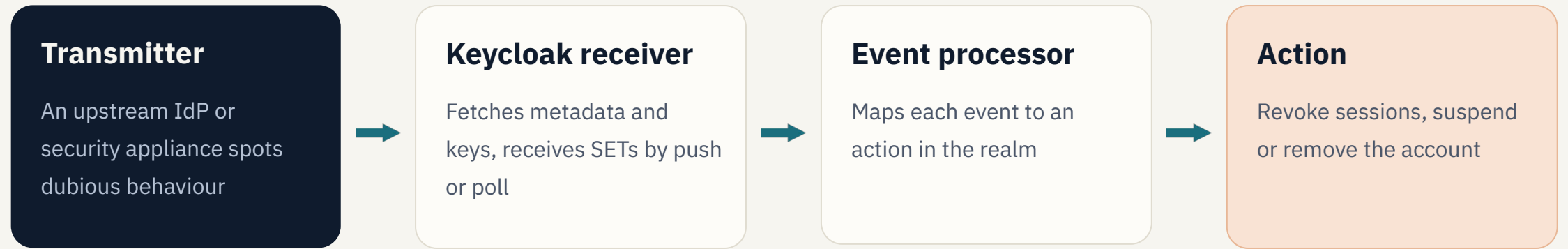
```
:26.7.5
```

```
:26.8.0
```

Inside the transmitter



Next: Keycloak as SSF receiver



TODAY

Proof of concept in [PR #43950](#)

DESIGN

Tentative — subject to change

NEXT

Receiver support coming to Keycloak soon

DEMO 2 OF 4

Keycloak as SSF receiver

PROOF OF CONCEPT

github.com/keycloak/keycloak/pull/43950

How to consume Shared Signals?



easyssf

easyssf.org

Shared Signals Framework support for Java

Spring Boot starter · Quarkus extension · plain Java library

OpenID SSF 1.0 · CAEP · RISC · SCIM events · tested with the OpenID conformance suite

DEMO 3 OF 4

Spring Boot SSF receiver

DEPENDENCY

```
org.easyssf:easyssf-receiver-spring-boot-starter:0.5.0
```

Add a receiver to any Spring Boot app

PUSH • RFC 8935

Accepts inbound SETs on a MVC route at /ssf/push

POLL • RFC 8936

Pulls SETs on a timer, or on demand via pollNow()

VERIFICATION

Checks every SET's signature against the JWKS, plus iss, iat, jti and aud

STREAM LIFECYCLE

Discovers the transmitter and manages its own stream

github.com/easyssf/easyssf • [easyssf-receiver-spring-boot-examples](https://github.com/easyssf/easyssf/tree/master/spring-boot-examples) • easyssf.org/#start-spring-boot

DEMO 3 OF 4

Quarkus SSF receiver

DEPENDENCY

```
io.quarkiverse.openid-ssf:quarkus-openid-ssf-receiver:0.5.0
```

Add a receiver to any Quarkus app

PUSH • RFC 8935

Accepts inbound SETs on a Vert.x route at /ssf/push

POLL • RFC 8936

Pulls SETs on a timer, or on demand via pollNow()

VERIFICATION

Checks every SET's signature against the JWKS, plus iss, iat, jti and aud

STREAM LIFECYCLE

Discovers the transmitter and manages its own stream

github.com/quarkiverse/quarkus-openid-ssf • [example app](#) • easysf.org/#start-quarkus

05 · WRAP-UP

One more thing...

Identity federation with Apple

Use Keycloak as the identity provider for Apple Accounts.

ABM

Apple Business Manager

ASM

Apple School Manager

Another step towards digital sovereignty — self-hosted identity for managed Apple devices. (SSF required for ABM/ASM integration)

[Keycloak / Apple Business Manager integration instructions](#)



Einstellungen

Suchen



Apple Account

Melde dich an, um auf deine iCloud-Daten, den App Store, Apple-Dienste und mehr zuzugreifen.



Konfiguration abschließen

1



Flugmodus



WLAN

DL5GHZ >



Bluetooth

Ein >



Mobilfunk



Batterie



Allgemein



Bedienungshilfen



Anzeige & Helligkeit



Barrierefreiheit



Key takeaways

- 01 SSF standardizes real-time security event exchange between identity systems.
- 02 Events travel as signed SETs; CAEP and RISC, SCIM Events define what they mean.
- 03 Push or poll delivery, organized in receiver-managed streams.
- 04 SSF transmitter support is planned for Keycloak 27.0.0 (experimental); receiver support is coming soon.
- 05 Please try it and share feedback on the SSF transmitter support in Keycloak. ([Discussion / Feedback](#))

[SSF Transmitter as a Preview Feature #49254](#)

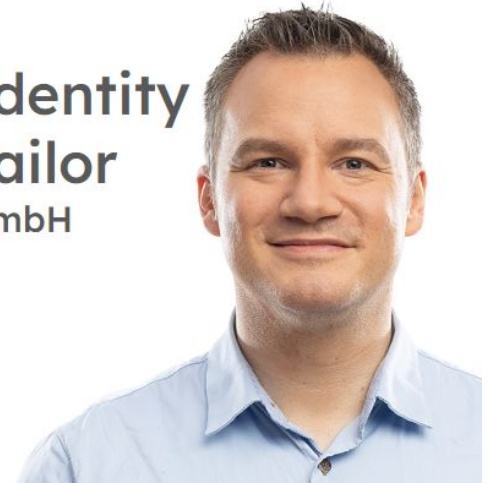
THANK YOU

Questions? Feedback?

thomas.darimont@identity-tailor.de

in/thomasdarimont · @thomasdarimont

Identity
Tailor
GmbH



 **Shared Signals Working Group**
OpenID Foundation

Please join! openid.net/wg/sharedsignals