Die Vitamine: Nutzen

Zuzana Trubini

Cnlab Herbsttagung
KOSMOS, Zürich, 7. September 2022

cnlab

Die Vitamine und Schadstoffe

Zuzana Trubini

Cnlab Herbsttagung
KOSMOS, Zürich, 7. September 2022

cnlab

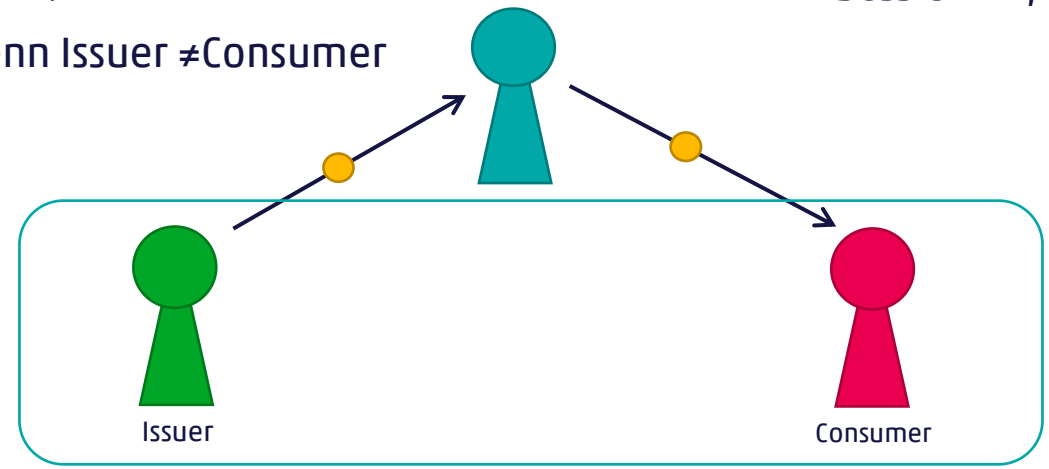
Token & Informationsgehalt


```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022,
  "iss": "auth-server"
}
```


- Self-contained Token (Assertion)
  - enthält strukturierte Daten: typischerweise Issuer, Audience, Gültigkeitsdauer, Informationen über den Enduser und den Client
  - meistens signiert, manchmal verschlüsselt
  - typischerweise direkt lokal validierbar (ohne State und Kommunikation)
  - besonders geeignet wenn Issuer ≠Consumer
  - z.B. JWT


```
sid=edb0e8665db4e9042fe0176a89aade16
```


- Opaque Token (Handle)
  - kein Informationsgehalt
  - Referenz auf eine interne Datenstruktur
  - zum Validieren ist State (oder Kommunikation) nötig
  - besonders geeignet wenn Issuer = Consumer
  - z.B. Session-ID, Refresh Token



Self-contained Token – Semantik

Semantik

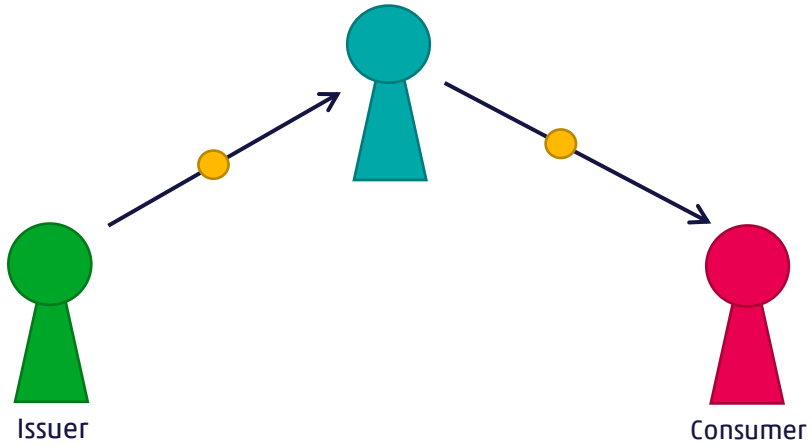
- Nachricht vom Issuer für Consumer mit einer Aussage über ein Subjekt, z.B. :

  1. «Der Überbringer dieser Nachricht hat sich als JD authentisiert.»

  2. «JD hat sich authentisiert.»

- Typisch für ID-Token (als Resultat von OIDC)
- Bearer Token (Überbringer = Bearer)
- Kann von jedem verwendet werden

- Keine Aussage über den Überbringer
- Internes Token



Self-contained Token – Syntax & Semantik

Semantik

Auth-Server Applikationsserver Endbenutzer

- Nachricht vom Issuer für Consumer mit einer Aussage über ein Subjekt, z.B. :
  1. «Der Überbringer dieser Nachricht hat sich als JD authentisiert.»
  2. «JD hat sich authentisiert.»

Syntax

- Issuer: Auth-Server
- Audience: Applikationsserver
- Subject: JD

Diametral unterschiedliche Bedeutung bei (fast) identischem Aussehen
Hohe Verwechslungsgefahr

Token & Sessionmanagement

- Session Start – Login: Passwort für ein Session-Token

«Der Überbringer dieses Token ist JD.»

- Requests mit Session-Token

Vom Client mitgeschickt und vom Server geprüft

  - Session-ID – nichtssagender zufälliger Bitstring, Bedeutung im Session-Kontext

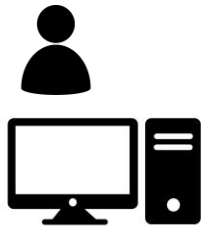
Validierbar nur mit State oder Kommunikation

  - JWT – Self-contained Token

Lokal und stateless prüfbar

- Session Ende – Logout, Time-out, Session-Kill

Sessionmanagement klassisch – mit Session-ID und State



Login
- User authentisiert sich
- Client schickt Credentials an Server

- Client speichert Session-ID

Logout
- User loggt sich aus
- Client schickt Logut-Request an Server

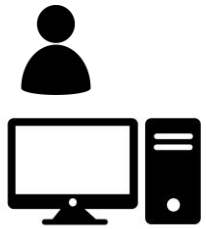
Session Start
- Server generiert & speichert Session-ID mit Session-Kontext
- Server schickt Session-ID an Client

Session-ID wertlos

Session Ende
- Server löscht Session-ID

Möglich auch bei Time-Out oder Session-Kill

Sessionmanagement ohne State (mit Self-contained Token)



Login
- User authentisiert sich
- Client schickt Credentials an Server
- Client speichert JWT

Logout
- User loggt sich aus
- Client löscht das JWT

Der Benutzer kann nichts mehr tun

Wie lang darf dieser Zustand sein?

Der Angreifer kann nichts mehr tun

Session Start
- Server generiert JWT & schickt es an Client

Session Ende
- JWT ist abgelaufen

Kein serverseitiges Logout
Kein Session-Kill

Session Management ohne State: Dieses Versprechen kann man fast nicht halten.

Sessionmanagement mit Self-contained Token und State

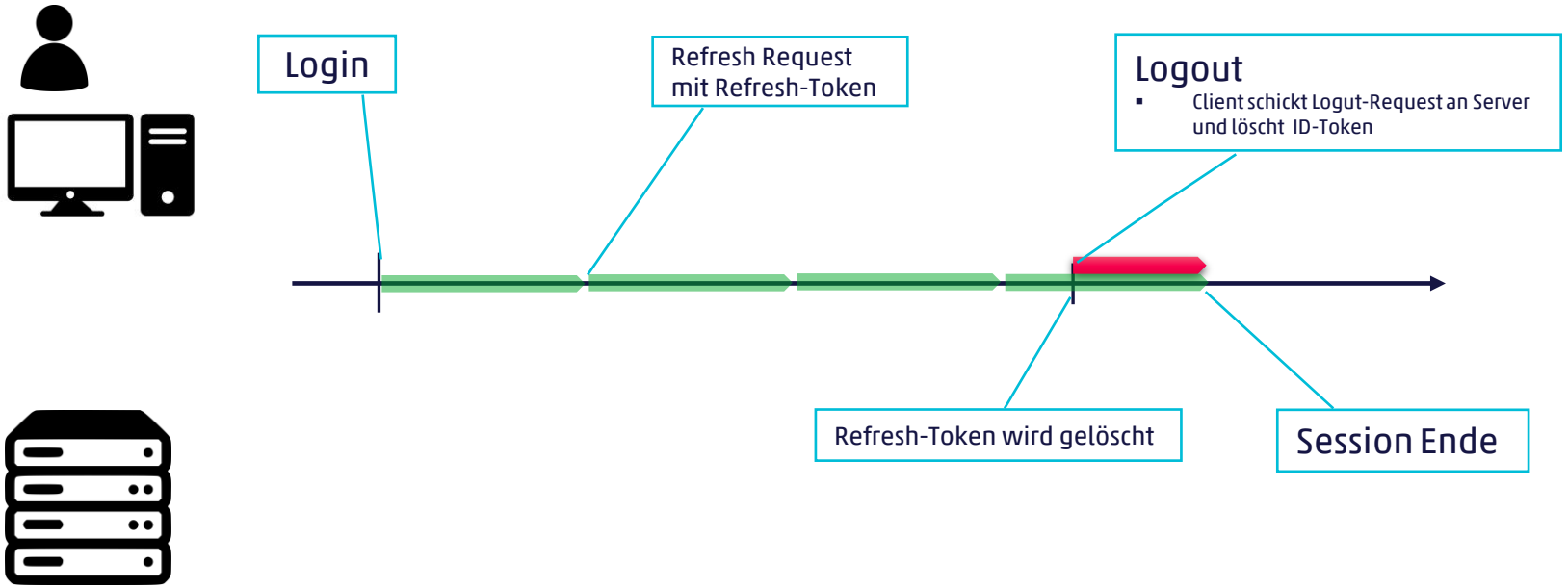
- Langlebiges ID-Token & Revokation-Liste

Blacklist-Prüfung bei jedem Zugriff

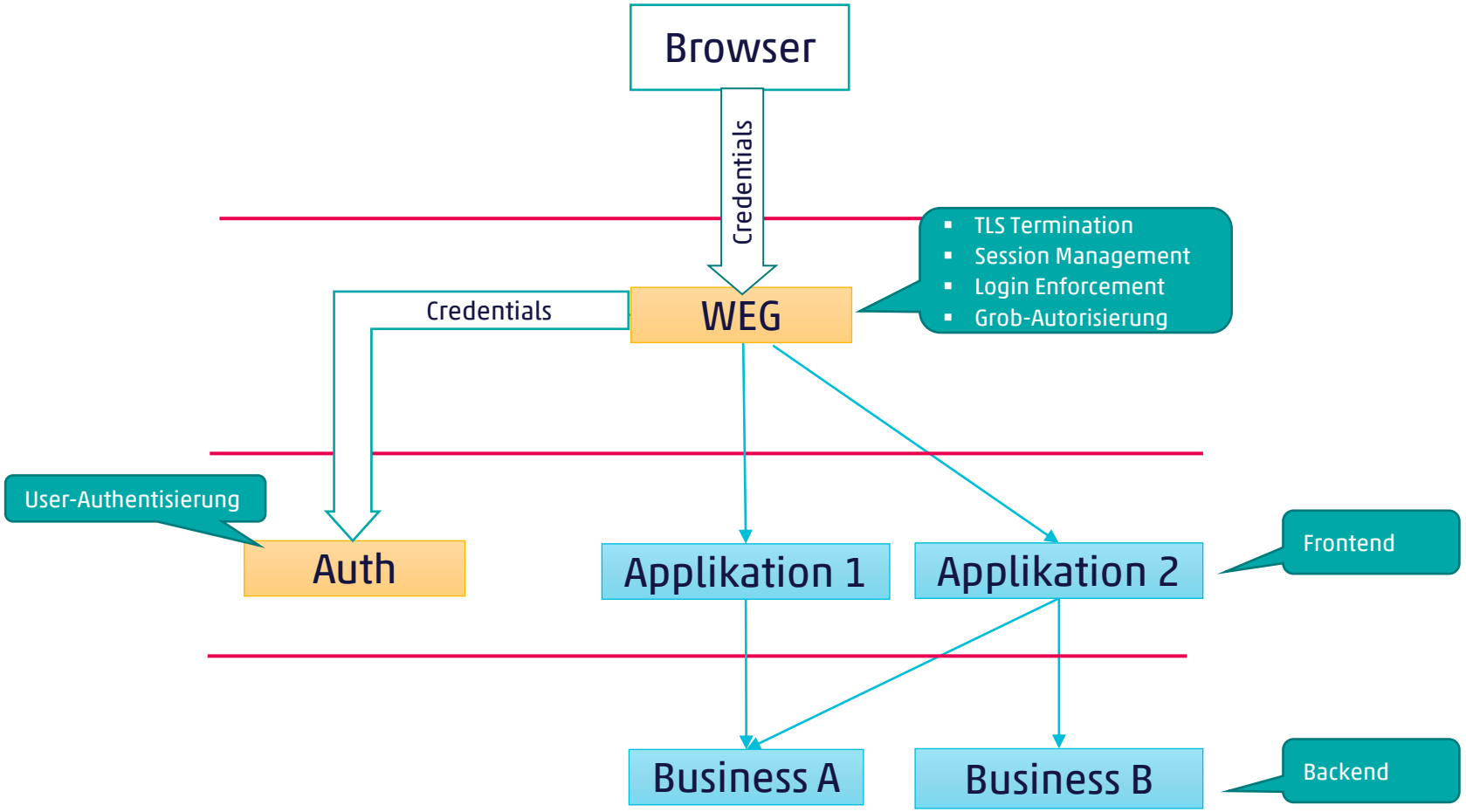
- Kurzlebiges ID-Token & revozierbares Refresh-Token

- Für «normale» Requests
- Lokal und stateless prüfbar

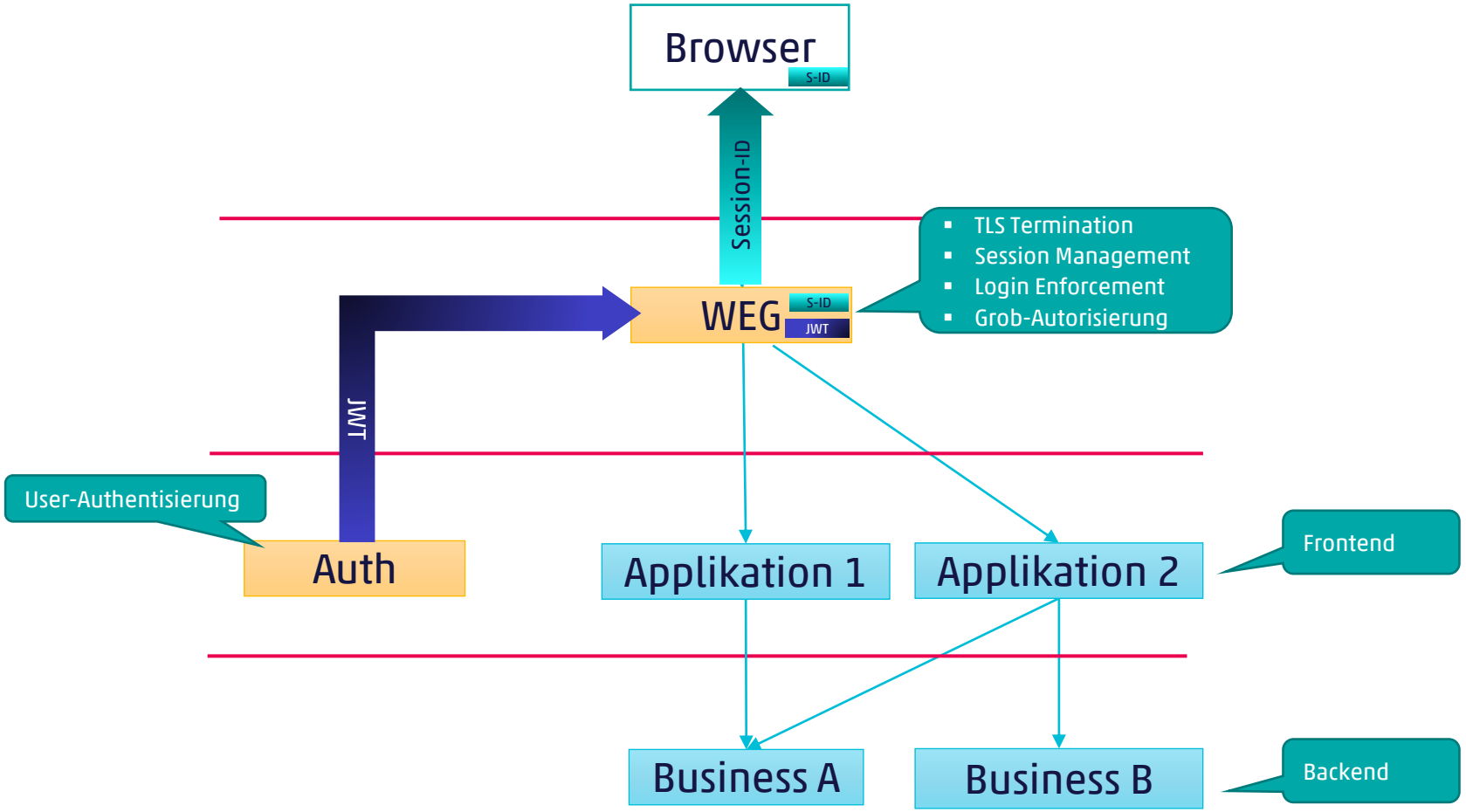
- Fürs Refreshen vom ID-Token
- Whitelist-Prüfung beim Refresh



Authentisierung & Autorisierung (interne Sicht)

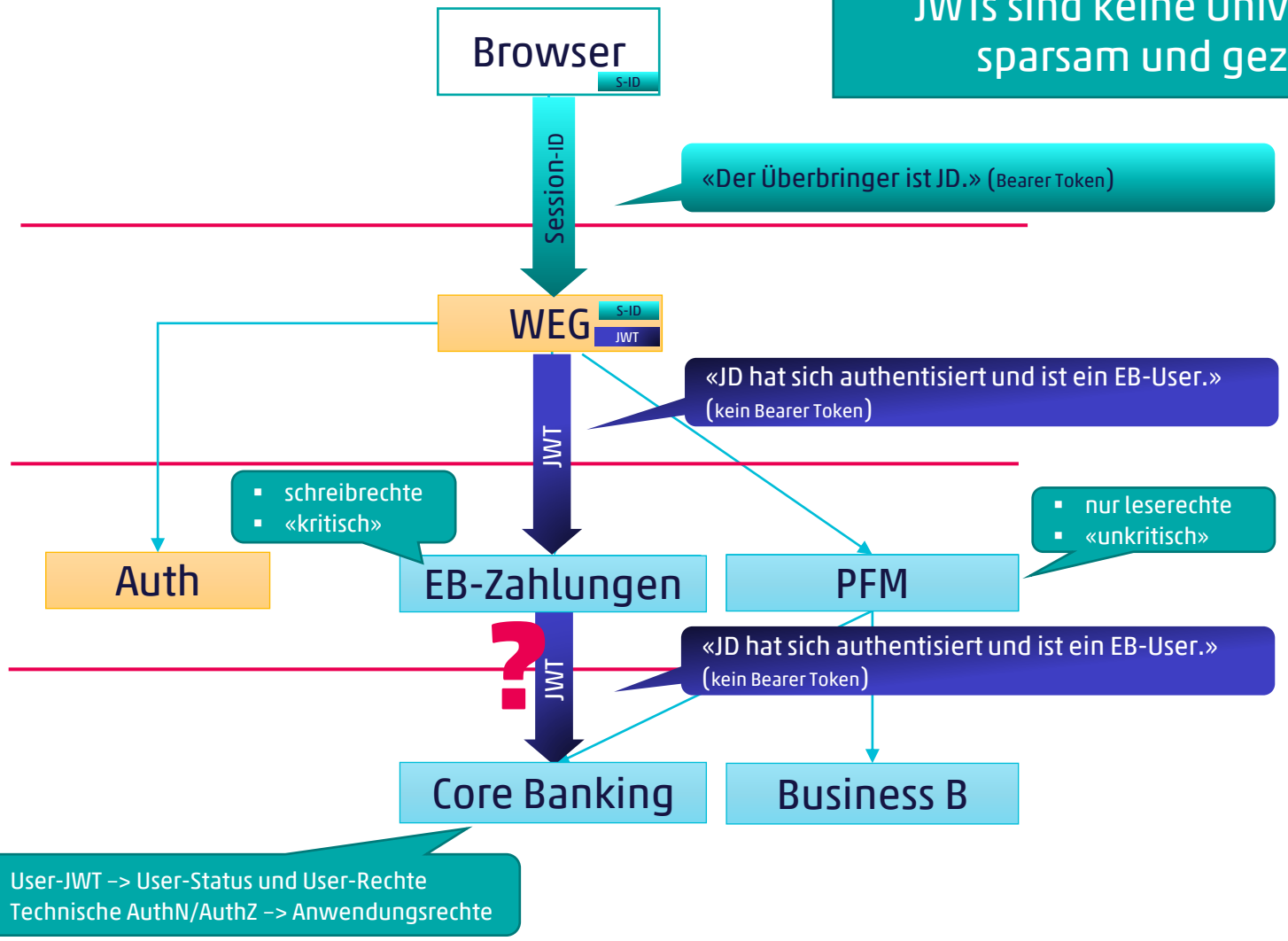


Authentisierung & Autorisierung (interne Sicht)

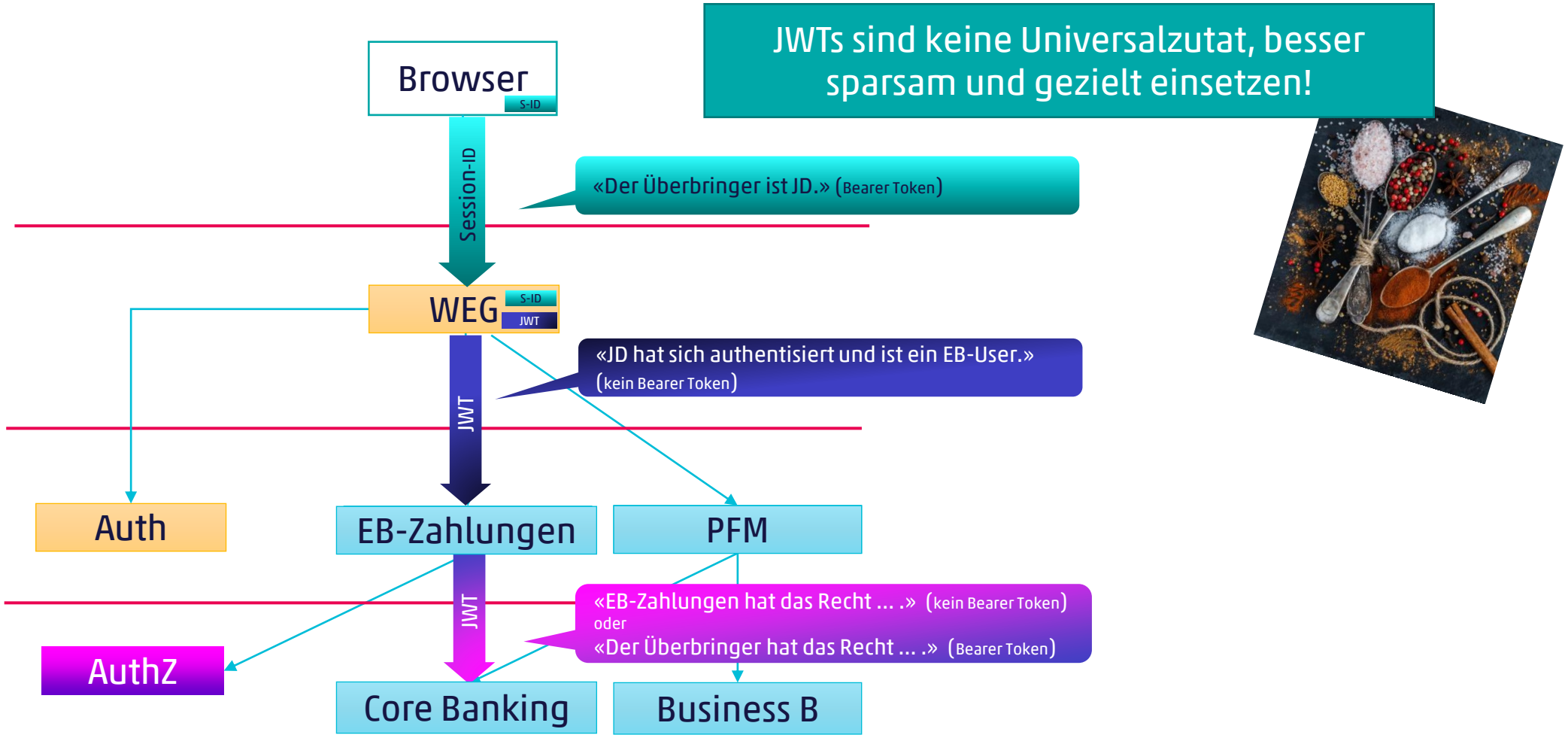


Authentisierung & Autorisierung (interne Sicht)

JWTs sind keine Universalzutat, besser sparsam und gezielt einsetzen!



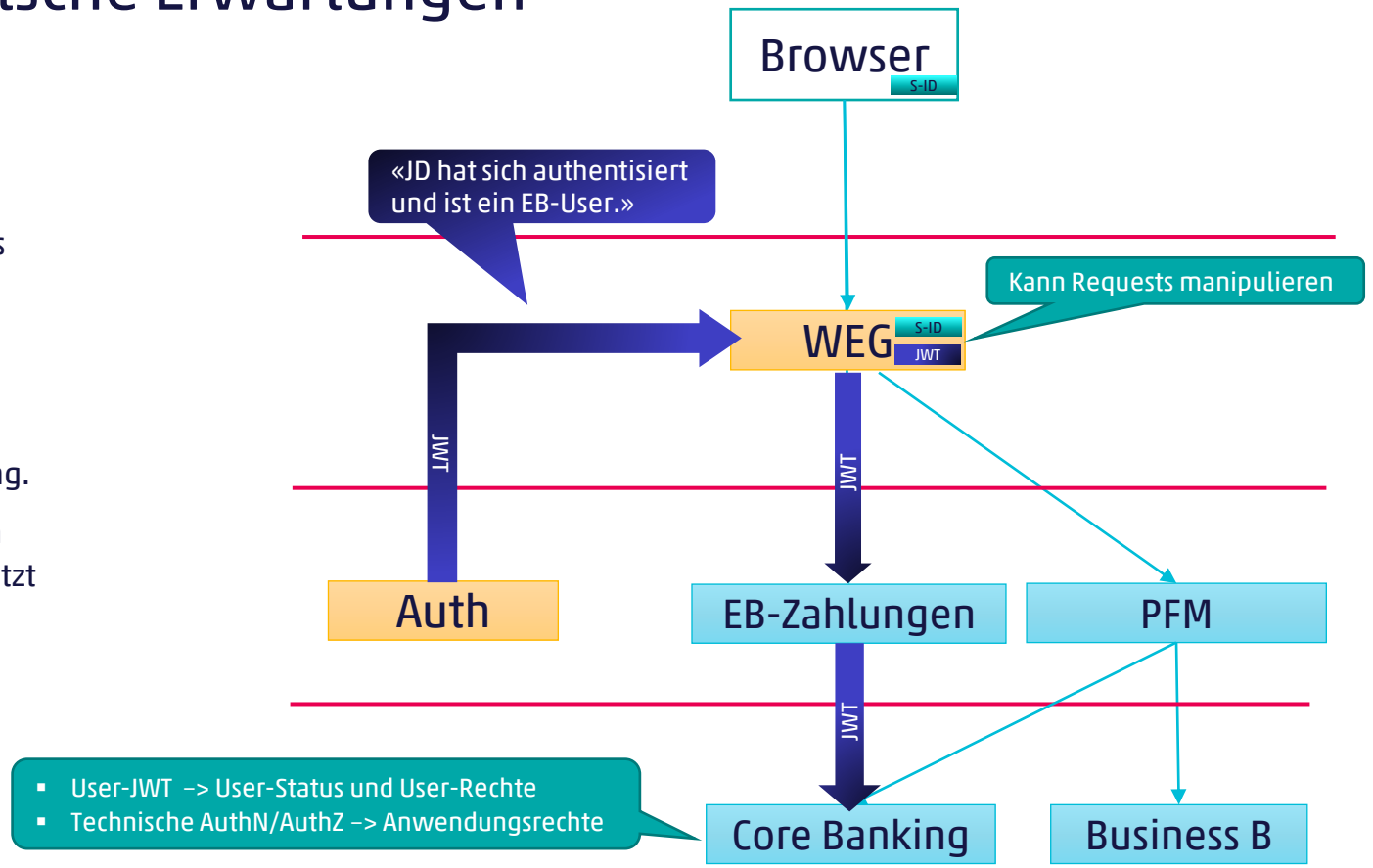
Authentisierung & Autorisierung (interne Sicht)



Token, Zero-Trust und falsche Erwartungen

Durch User-Token Prüfung kann eine Anwendung sicherstellen, dass sie nur Requests für kürzlich authentisierte EB-User entgegennimmt, aber:

- Ein gültiges User-Token garantiert nicht, dass ein Request tatsächlich vom User stammt. Diesbezüglich **vertraut** man dem WEG (und jedem anderen «Zwischensystem»).
- Eine User-Token Prüfung ist **kein Ersatz** für technische Authentisierung und Autorisierung.
- Beim Einsatz von Bearer Token **vertraut** man jedem Empänger, dass er das Token gut schützt und nicht missbraucht.



Token Prüfen ≠ Zero-Trust Garantie

Fazit

- Egal was Sie kochen, erstellen Sie immer zuerst ein Rezept (Konzept).
- Überlegen Sie, was das Ziel ist und welche Zutaten (Token) dafür geeignet sind.
- Spezifizieren Sie die Bedeutung der verwendeten Token.
- Setzen Sie Self-contained Token gezielt ein.
- Setzen Sie Bearer Token sparsam ein.
- Token prüfen ≠ Zero-Trust Garantie – analysieren Sie, wem Sie wo vertrauen.

Vielen Dank für Ihre Aufmerksamkeit_

info@cnlab-security.ch
+41 55 214 33 40

cnlab security AG
Obere Bahnhofstrasse 32b
CH-8640 Rapperswil-Jona
Switzerland

cnlab