VIDIS

(Testsystem) Whitepaper zur Anbindung an VIDIS für Service Provider

Version: 2
Exportiert am 12-01-2023

Das Medieninstitut der Länder

Inhaltsverzeichnis

1. Informationen für Service Provider (SP)

Um die geordnete Anbindung eines "Service Providers" an das VIDIS-System zu ermöglichen, müssen entsprechende Metadaten gemäß https://openid.net/specs/openid-connect-discovery-1_0.html ausgetauscht werden.

In diesem Dokument wird ausschließlich auf den Fall einer Anbindung von Angeboten (Webseiten und Apps) als OpenID Connect Client eingegangen.

Es werden in späteren Projektphasen noch weitere SSO-Technologien für den Einsatz evaluiert. Es ist jedoch unwahrscheinlich, dass der produktive VIDIS-Dienst später viele weitere SSO-Technologien unterstützen wird.

2. Demosystem / Proof of Concept (PoC) / Pilotsystem

Ein Demosystem / Proof of Concept (PoC) ist derzeit in Betrieb. Alle Tests, insbesondere die Test-Anbindungen, werden derzeit an diesem VIDIS-System durchgeführt. Dieses Demosystem wird in Zukunft dauerhaft als Integrations- und Testsystem für die VIDIS-Infrastruktur dienen und eine Anbindung wird Voraussetzung für die Anbindung an das Pilotsystem sein. Das Pilotsystem wird derzeit aufgebaut, verhält sich analog und die Inbetriebnahme ist für Ende Februar geplant.

2.1. Anbindung eines OpenID Connect Client

Für die Anbindung eines OpenID Connect Clients müssen folgende Parameter ausgetauscht und sowohl im VIDIS-System als auch beim anzubindenden OpenID Connect Client konfiguriert werden.

2.1.1. FWU/VIDIS an SP zusammengefasst

- **"ClientID":** der Identifikator des anzubindenden OpenID Connect Clients
- **"Client Secret"**: ein zwischen VIDIS und dem OpenID Connect Client geteiltes Geheimnis
- **"Authorize Endpoint":** https://aai-test.vidis.schule/auth/realms/vidis/protocol/openid-connect/auth
- **"Access Token Endpoint":** https://aai-test.vidis.schule/auth/realms/vidis/protocol/openid-connect/token

Die maschinenlesbare OpenID Connect Konfiguration befindet sich zusammengefasst unter: https://aai-test.vidis.schule/auth/realms/vidis/.well-known/openid-configuration

2.1.2. SP an FWU/VIDIS zusammengefasst

Notwendig:

- **"Valid Redirect URIs":** zulässige Redirect URIs
- **"BaseURL":** zum testen der Verbindung von FWU-Seite (Das ist der Einstiegspunkt bzw. das Login Formular)
- **"Deeplink" ins Angebot:** Wird benötigt um von Landesportalen mit Session direkt in Ihr Angebot zu springen ohne erneute Anmeldung

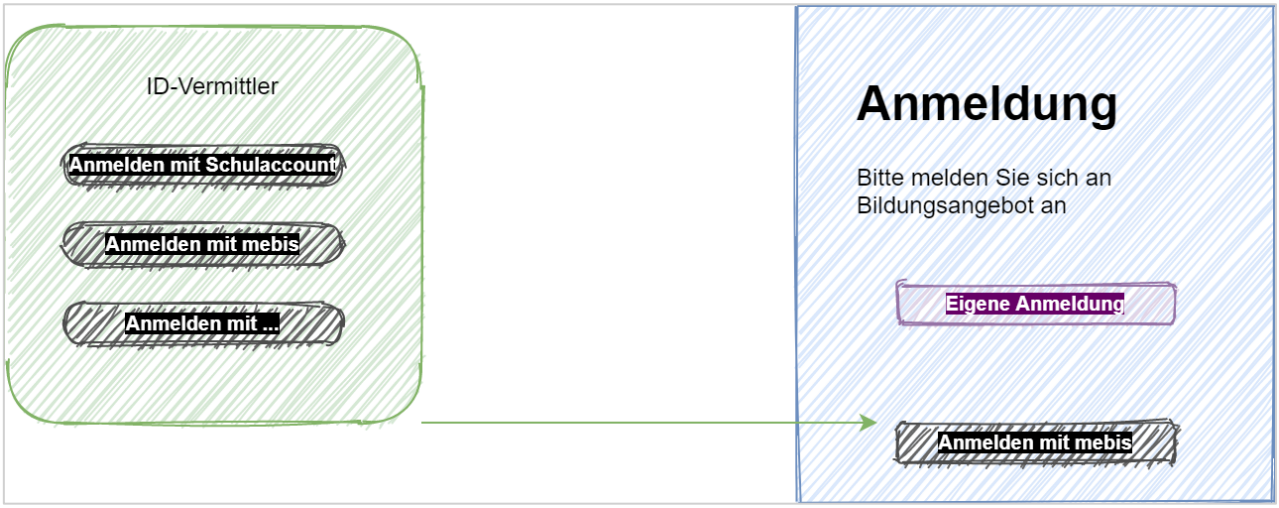
- Weitere technische Voraussetzungen siehe unten

Optional:

- Social-Media-Vorschaubild für Deeplink: Hilfreich bei Verlinkung bzw. Anzeige in Lern-Management-Systemen, Mediatheken und Portalen

2.2. Integration VIDIS Button

VIDIS bietet eine (JavaScript) Client-Bibliothek an, die von Bildungsangeboten eingebunden werden kann. Sie hat zum Ziel, die Usability deutlich zu verbessern, ohne zu stark in das User-Interface der Bildungsangebote einzugreifen. Eine Teilnahme an VIDIS ist auch ohne diese Client-Bibliothek möglich. Allerdings ist es wünschenswert den Button beispielsweise auf einer Testseite (private URL) einzubinden.



Code zur Einbindung auf Anmeldeseite

3. Wie integriert man den Vidis-Login?

3.1. Was ist der Vidis Login?

Das Ziel des Vidis-Logins ist es, jedem Schüler und Lehrer ein einziges Konto zur Verfügung zu stellen, mit dem man sich überall einloggen kann.

Man könnte hier an das Google Login denken, aber anstelle eines Google-Kontos wird ein Schul-Login als IDP verwendet. Das bedeutet, dass der Vidis Login den Benutzer in das IDP-System einloggt, das die Schule verwendet und zu Ihnen zurückkehrt.

Der Benutzer muss also die IDP auswählen, die er verwenden kann, was über den Vidis Login Button geschehen kann.

Sie können die Funktionalität hier testen: https://tp.fwu.intension.eu/?version=latest

3.2. Voraussetzungen für die Integration des Vidis-Login:

- Unterstützung Openid-Connect

3.3. Integration des Vidis Login Buttons:

3.3.1. Allgemeine Vorgehensweise:

1. Hinzufügen eines CDN-Links
2. Bauen Sie die Webkomponente des Vidis Login-Buttons ein
3. Geben Sie dem Button Ihren Login-Link
4. Konfiguration des Vidis Login-Buttons
    a. Größe (size)
    b. Cookie (cookie)

3.3.2. 1. Hinzufügen eines CDN-Links

Damit der Vidis Login Button funktioniert, müssen Sie einen CDN-Link in Ihre Website einbinden:

```
<script src="https://repo.vidis.schule/repository/vidis-cdn/latest/vidisLogin.umd.js"></script>
```

Der Link, den Sie normalerweise verwenden würden,
lautet: `https://repo.vidis.schule/repository/vidis-cdn/latest/vidisLogin.umd.js`

Wenn Sie eine bestimmte Version bevorzugen: `https://repo.vidis.schule`/repository/vidis-cdn/{version}/vidisLogin.umd.js

Zum Beispiel: `https://repo.vidis.schule`/repository/vidis-cdn/0.11.0/vidisLogin.umd.js

3.3.3. 2. Bauen Sie die Webkomponente des Vidis Login-Buttons ein

Wenn Sie sich entschieden haben, wo der Vidis-Login-Button platziert werden soll, können Sie ihn wie folgt hinzufügen:

```
<vidis-login loginurl=""></vidis-login>
```

3.3.4. 3. Geben Sie den Login-Link für den Button an

WICHTIG: Sie müssen die Loginurl Ihres Systems angeben, sonst kann der Button nicht funktionieren.

WICHTIG: Die Loginurl muss mit "https://" beginnen, um erkannt zu werden.

Der Benutzer wird umgeleitet, indem die angegebene Loginurl mit diesem Abfrageparameter ergänzt wird: kc_idp_hint

3.3.5. 4. Konfiguration des Vidis Login Buttons

Sie können den Vidis Login Button mit den folgenden Attributen anpassen:

- Größe: Bestimmt die Größe des Buttons.
    - Werte:

- "L": Groß, zeigt die Schaltfläche in einer großen Version an. Dies ist auch die Standardeinstellung.
- "M": Mittel, zeigt die Schaltfläche in einer mittleren Version an.
- "S": Klein, zeigt die Schaltfläche in einer kleinen Version an.

- Cookie: Aktiviert oder deaktiviert die Speicherung der letzten Auswahl des Benutzers in einem Cookie.
  - Es wird empfohlen, diese Option zunächst auf "false" zu setzen und erst dann zu aktivieren, wenn der Benutzer den Cookies auf Ihrer Website zugestimmt hat, um rechtliche Probleme zu vermeiden.

3.3.6. Ein vollständiges Beispiel:

Vidis-Login Example

```
<script src="https://repo.vidis.schule/repository/vidis-
cdn/1.0.1/vidisLogin.umd.js"></script>
...
<vidis-login loginurl="https://www.domain.de/path-to-
auth/" size="L" cookie="true"></vidis-login>
```

3.3.7. Kompatibilität:

Der Vidis Login Button ist als Webkomponente erstellt und sollte daher in jeder html-basierten Umgebung funktionieren, insbesondere in jedem SPI-Framework wie Vue, Angular und React.

Technisch gesehen, wenn man weiß, wie man eine Webkomponente in andere Apps (wie Android oder IOS) integriert, sollte der Button auch out of the box funktionieren, ist aber noch nicht dafür getestet.

4. Weitere technische Voraussetzungen

4.1. Automatische Registrierung bei initialer Anmeldung

Voraussetzung für Service Provider ist, dass Nutzerinnen und Nutzer, die sich an dem digitalen Bildungsangebot erstmalig anmelden, bei der Anmeldung automatisch registriert werden.

4.2. Weitergabe des Parameter vidis_idp_hint

Voraussetzung für Service Provider ist, die automatische Weitergabe des Parameter vidis_idp_hint (z.B. "?vidis_idp_hint=Landessystem") während der Anmeldung. Für einige Anwendungsfälle (z.B. Direktaufruf aus Landessystemen heraus) ist die Vorauswahl eines Landes-IdP wichtig. Der VIDIS-Dienst unterstützt diese Vorauswahl durch Übergabe des Parameter vidis_idp_hint. Dieser Parameter muss also bei aufrufen (z.B. bei Authorization Requests) uneditiert weitergegeben werden und darf nicht herausgefiltert werden.

5. Testen Ihrer Anbindung

Zum testen Ihrer Anbindung rufen Sie Ihre "**Callback-URL**" auf. Das sollte Sie zum Login mit Vidis leiten.

Dort müssen Sie das **Test-Landesportal(IdP)** auswählen und sich mit unserem Test Account einloggen, danach sollten dann bei erfolgreichem Login in Ihrem System authentifiziert sein und Zugriff auf alle freigegebenen Inhalte haben.

5.1. Test Account Daten:

Username: fwu-testuser

Passwort: das Passwort erhalten erhalten Sie separat

5.2. Userinfo überprüfen

Über den endpoint https://aai-test.vidis.schule/auth/realms/vidis/protocol/openid-connect/userinfo können Sie sich die Übermittelten Userdaten unsererseits ansehen.

Das funktioniert wie folgt.

Schritt 1: Zuerst muss ein Token generiert werden.

Dieser Token (access_token) fungiert als "Bearer Token" der zum Abruf der User Info genutzt werden kann und zu Debugging zwecken, der Redirect Flow wird bei der finalen Integration verwendet.

```
curl --location --request POST 'https://aai-
test.vidis.schule/auth/realms/vidis/protocol/openid-connect/userinfo' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'username=fwu-testuser' \
--data-urlencode 'password=XXXXX' \
--data-urlencode 'grant_type=password' \
--data-urlencode 'client_id=account'
```

Schritt 2: Jetzt lassen sich die Userdaten mit dem Token abrufen

```
curl --location --request GET 'https://aai-
test.vidis.schule/auth/realms/vidis/protocol/openid-connect/userinfo' \
--header 'Authorization: Bearer <TOKEN>
```

oder alternative zu Schritt 2 wäre es den JWT Token zu dekodieren, um die Nutzerdaten auslesen zu können. Hierzu kann beispielsweise auch ein entsprechendes Online Tool verwendet werden (z. B. über https://devtoolzone.com/decoder/jwt, https://jwt.io/ etc.), sofern es sich nicht um reale Nutzerdaten handelt.

Per Code kann ein JWT Token beispielsweise wie folgt dekodiert werden:

```
import org.Apache.commons.codec.binary.Base64;
@Test
    public void testDecodeJWT(){
        String jwtToken =
"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJ0ZXN0Iiwicm9sZXMiOiJST0xFX0FETUlOIiwiaXNzIjoibXlzZW
xmIiwiZXhwIjoxNDcxMDg2MzgxfQ.1EI2haSz9aMsHjFUXNVz2Z4mtC0nMdZo6bo3-x-aRpw";

        String base64EncodedHeader = split_string[0];
        String base64EncodedBody = split_string[1];
        String base64EncodedSignature = split_string[2];

        Base64 base64Url = new Base64(true);
        String header = new String(base64Url.decode(base64EncodedHeader));
        System.out.println("JWT Header : " + header);
        String body = new String(base64Url.decode(base64EncodedBody));
        System.out.println("JWT Body : "+body);
    }
```

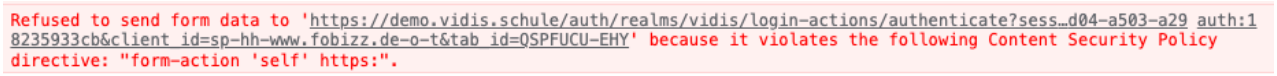
Später im Livebetrieb werden die Userdaten nicht mehr in dem *access_token* stehen.

Das Ergebnis ist ein JSON mit den Userdaten:


```
{
  "sub": "39e0063a-8377-4a1c-bd15-ea16f65a5a15",
}
```

6. Mögliche Fehler

6.1. Form Action



Der aktuelle Header für "form-action" ist "'self' https:" ist so konfiguriert, dass grundsätzlich damit jede HTTPS-Seite funktionieren sollte. In dieser Fehlermeldung wird kritisch angemerkt, dass im aktuellen Betrieb des PoC, die Verschlüsselung zwischen Host und lokaler Testinstanz fehlt. Dies wird in den späteren Umgebungen (z.B. Pilotumgebung) kein Problem darstellen, da diese SSL verschlüsselt sein werden. Es betrifft nur die lokale Entwicklung. Nach außen (externe Verbindungen außerhalb des Rechenzentrums) findet eine korrekte Verschlüsselung per https statt.

7. Fragen & Anregungen

Für Fragen und Anregungen melden Sie sich gerne jederzeit unter vidis@fwu.de