ChatGPT

SPIRAL ver.1におけるOkta認証可否の調査報告

以下、SPIRAL　ver.1上でOktaによる認証（シングルサインオン）を実現できるかについて、現時点での調査結果を整理します。

1. 想定するOkta認証フローと適用範囲

現在想定しているのは、**エンドユーザー向けWebサイト（SPIRALで構築した会員サイト）のログイン時にOkta認証を挟む**フローです。具体的には、ユーザーがSPIRALサイトにログインしようとしたタイミングで外部IdPであるOkta（社内ADと連携済み）による認証を行い、その結果に基づいてSPIRAL側のユーザー識別・アクセス制御を行う形を検討しています。

- **適用対象:** まずはエンドユーザー用の会員サイトのログイン機能にOkta認証を組み込むことを目指します。
- **認証タイミング:** ユーザーがSPIRALサイトのログインページにアクセスした際に、SPIRAL側の認証処理に入る前にOktaでのログイン（シングルサインオン）を実施します。
- **処理イメージ:** ユーザーが「ログイン」ボタンを押すとOktaの認証ページにリダイレクトし、Oktaで社内資格情報による認証が成功したらSPIRALサイトに戻って自動ログイン完了（該当ユーザーのマイページ表示）という流れを想定しています。認証成功後は、Oktaから受け取ったユーザー識別情報をもとにSPIRAL内の会員データベースと照合し、対応するユーザーのセッションを確立する形です。

※今回の検討範囲はあくまでエンドユーザーサイトでのシングルサインオン導入可否です（SPIRAL管理画面へのSSO適用は、まずエンドユーザー側での実現性を確認してから別途検討します）。

2. SPIRAL ver.1のカスタムPHP機能と制約

SPIRAL ver.1では、ページ内に埋め込む形でPHPコードを実行することが可能ですが、**利用できる関数や機能にいくつか制約があります**。Okta連携の実装可否を判断するため、関連しそうなポイントを整理します。

- **ファイル操作・システムへの影響がある関数は使用不可:** セキュリティ上の理由から、ファイルシステムへ影響を及ぼす可能性のあるPHP関数・クラスはSPIRAL環境で無効化されています [1]。例えば、ファイル入出力系（`fopen`, `file_get_contents`, `file_put_contents`など）やソケット通信系（`fsockopen`など）は使用不可です [2] [3]。したがって、外部サービスとの通信を行う際にも、直接`file_get_contents('https://....')`で呼び出すことはできません。

- **cURLによる外部HTTP通信は可能:** 上記の制限はありますが、PHP標準のcURLライブラリは利用可能です。実際、スパイラルのナレッジサイトにおいても、PHPカスタムプログラムから**外部のWeb　APIを呼び出すサンプル**（Google MeetやZoomの会議URLを自動発行する例）が紹介されており、その中で`curl_init()`や`curl_exec()`を用いたHTTP通信が行われています [4] [5]。このことから、SPIRAL　ver.1環境でもHTTPS経由でOktaの認可エンドポイントやトークンエンドポイントに接続すること自体は技術的に可能と考えられます。ネットワーク経路の制限についても、「外部通信可否に留意」との注意書きがあるものの [4]、標準的なHTTPSであれば問題なく接続できると推測されます。

- **PHPセッション管理:** SPIRALのPHPでは通常の`$_SESSION`も利用可能です（SPIRAL提供の`$SPIRAL->getSession()`は内部的にPHPセッションをラップしており、セッションIDの再生成やタイムアウト時のクリア処理が自動化されています [6]）。`session_start()`自体も使用できますが、一部オプ

ション（`save_path`や`use_cookies`等）についてはセキュリティ上指定できないよう制限されています[7]。従って、**セッションを使った状態保持（例：Oktaから受け取ったトークン情報や一時的な`state`値の保存）は可能**です。

- **ヘッダ操作やリダイレクト:** カスタムPHPからHTTPヘッダを直接操作する関数（例えば`header()`によるリダイレクトやCookie手動送信）は禁止されています[8]。このため、Okta認証フローで必要となる**ユーザーのブラウザリダイレクト**をサーバー側PHPだけで完結させることはできません。代替策としては、PHP側で適切なHTMLやJavaScriptを出力し、クライアントサイドでOktaの認証URLへ遷移させる形になります（例：MetaタグやJavaScriptの`location.href`によるリダイレクト）。同様に、Okta認証後のコールバック（リダイレクトURI）から別のページへ遷移する場合も、サーバーで`header("Location: ...")`が使えないため、画面上でのリダイレクト処理が必要です。この点は実装上の工夫となります。

- **利用可能なライブラリ:** SPIRAL ver.1のPHP実行環境では、開発者が自由にComposerでパッケージを追加したり、独自のPHP拡張モジュールをインストールすることはできません。標準で有効な拡張（cURL, OpenSSL, JSON, MBStringなど）は使えますが、それ以外は**あらかじめ組み込まれている機能のみで実装する必要があります**。実際、SPIRAL提供の開発者向け関数を見ると、Facebook SDK用のヘルパー（`$SPIRAL->getFacebook()`）が用意されているなど、一部ライブラリは内包されていますが[9]、Okta用の専用ライブラリは当然ながら無いため、**OpenID Connectの処理を素のPHPで実装する必要**があります。今回ローカル環境で検証済みのとおり、「標準関数のみでOIDC認証フロー（Authorization Codeフロー）の実装」は可能なので、SPIRAL上でも同様のアプローチをとることになります。

- **暗号処理・トークン検証:** Oktaから返されるIDトークンはJWT形式ですが、PHP標準関数でJWTのデコード自体は`base64_decode`や`json_decode`で可能です。署名検証についてはOpenSSL機能の利用が考えられます。SPIRALではOpenSSL関連関数の一部がセキュリティ上禁止されていますが（鍵ファイルの読み込み/書き出し等）[10][11]、公開鍵での署名検証に使う`openssl_verify()`などはリストに挙がっておらず、使用できる可能性があります※。最悪、署名検証をスキップする手はありますが、セキュリティ担保のため可能であれば検証も行いたいところです。この点は**Oktaの公開鍵情報(JWKS)**を事前に取得し、`openssl_verify`でIDトークンの署名を検証する実装を検討しています（要確認ポイント）。

※JWT検証用のライブラリをインストールできないため、RSA署名の検証を行うにはOpenSSL関数に頼る必要があります。その際、`openssl_pkey_get_public`等の鍵取得関数は使用不可ですが[10]、証明書文字列から直接公開鍵リソースを生成する手段や、OktaのJWKSからモジュラスと公開指数を取得してOpenSSLキー構造体を生成する方法などで回避可能か検討します。

3. Okta連携方式の検討（OIDC vs SAML）

Oktaを外部認証基盤として用いる場合、一般的に**OIDC (OpenID Connect)** または **SAML** の2方式が候補になりますが、現状では **OIDC（OAuth2ベースのOpenID Connect）** を第一候補としています[12]。理由は以下の通りです。

- **OktaのOIDCサポートの充実:** OktaはOAuth2/OIDCによる認証を強力にサポートしており、ドキュメントやSDKも豊富です。OIDCはOAuth2.0を拡張した業界標準の認証レイヤーであり[12]、Okta上でクライアントアプリを作成してOIDC接続するのは比較的容易です。Authorization Codeフローを使うことでセキュリティ要件（`state`値によるリクエスト検証、`nonce`やPKCEによるコード不正利用対策など）も満たしやすく、信頼性の高いSSOが実現できます。

- **実装の柔軟性:** OIDCはRESTベースのAPI通信でトークンをやり取りする方式のため、今回のようにカスタムPHPでフローを実装するのに適しています。実際、前述したように**「ライブラリ非依存でPHPのみでOIDCフローを実装できる」ことはローカル検証済み**ですので、SPIRALの制約内で十分実装可能と考えています。将来的にアクセストークンを使ったAPI認可やユーザープロビジョニング連携などへ拡張する場合も、OIDC（＋OAuth2）の方が発展性があります。

- **SAML方式を採用しない理由:** SAML 2.0によるSSOも検討はしましたが、現時点では優先度を下げています。その理由は、**SPIRAL側でSAML連携を受け入れる仕組みが明確でない**ことと、SAML認証の実装がOIDCに比べて扱いづらい点です。例えば、OktaからSPIRALへのSAMLResponseを受信・検証してログインセッションを確立するには、SPIRALのページで独自にSAMLのXMLを処理する必要があります。しかしXML処理用のライブラリや関数制限、証明書検証の実装負荷を考えると、JSONベースで軽量なOIDCの方が適しています。また、Okta側でもSpiral向けに公式提供されている統合テンプレートは**SAML方式でのシングルサインオン（いわゆるSAML連携によるSSO設定）**となっています [13]。公式にOIDC連携が用意されていないということは、SPIRALがサービスプロバイダ(SP)としてSAMLなら連携実績がある一方、OIDCは標準機能としては持っていないことを示唆しています。この点からも、**あえて公式サポート外のOIDCを採用するのはカスタム実装前提のアプローチ**ではあります。しかし、SAML方式を無理に使うよりも、自前でOIDCフローを構築した方が実装・運用の自由度や拡張性で勝ると判断しました。

(補足: SPIRAL ver.1のマイエリア（会員機能）にはバージョン1.13でシングルサインオンの機能強化がアナウンスされており、SPIRALに登録された会員情報を他サービスの認証に利用できるようにするとされています [14]。これはどちらかというとSPIRALをIdPとして使うSAML連携のようにも読めますが、本件はSPIRALをサービス提供側として外部IdPに連携する話ですので、文脈が異なります。少なくとも現時点でSPIRALが外部IdP連携（受け手）を標準サポートしている情報は確認できませんでした。)

4. 現状の結論と今後の対応方針

以上の調査から、**SPIRAL　ver.1上でOkta認証（OIDC方式）を実装すること自体は「技術的に可能」である**と判断しています。大きな理由として、SPIRALのPHP実行環境で必要となる以下の要件が概ね満たされているためです。

- **外部HTTPS通信:** cURLを用いたOkta APIエンドポイントとの通信が可能である [5]。社内AD連携したOktaドメインへのアクセスもHTTPSであれば問題ないと思われます。
- **セッション等の状態管理:** PHPセッションを利用して一時的な認証コードやトークン、ユーザー情報の保持が可能である [6]。
- **JSON/Webトークン処理:** IDトークン（JWT）のデコードや、必要に応じた署名検証もPHP組み込み関数で対応可能な範囲である（OpenSSLの制約はあるものの代替手段検討可）。
- **画面遷移の制御:** サーバーサイドでの自動リダイレクトはできないものの、フロント側でのリダイレクト処理を組み込むことでOkta認証画面への誘導および認証後のコールバック処理を実現できる。
- **SPIRAL固有機能との両立:** Okta認証結果を受けてSPIRALの会員データベースを参照し、該当ユーザーのセッションを開始する処理はカスタムPHPとSPIRAL API（会員検索やセッション確立）で実装可能と思われます。SPIRAL提供の `$SPIRAL->getDataBase()` で会員DBを検索し [15]、一致するユーザーがいれば `$SPIRAL->getSession()->put()` でログイン情報をセッションに保存する、といった流れが考えられます。

一方で、**実装上注意すべきポイントや不明点**もいくつか残っています。

- **SPIRALセッションとログイン状態の紐付け:** SPIRALのマイエリア機能における「ログイン済み」状態をカスタムPHPからどこまで制御できるかは要確認です。単にセッションにユーザーIDを入れるだけで会員ページへアクセス可能になるのか、あるいはログインフォームを経由しない場合に備えて特別な取扱いが必要か（例えばSPIRAL標準のセッションCookieや認証用トークンを適切に発行する必要

があるか）を検証する必要があります。公式ドキュメントにはそのあたりの直接的な解説が無いため、テストしながら確認していく想定です。

- **リダイレクト処理のユーザー体験:**　フロント側でのリダイレクト実装になるため、画面遷移時に一瞬ユーザーにわかる形で画面が切り替わる可能性があります（例えばフォーム送信→Oktaログインページ→再度戻ってくる流れで、一瞬SPIRALの中間ページが表示される等）。可能な限りシームレスに見えるよう、画面デザインやスクリプトの工夫も必要でしょう。

- **セキュリティ上の考慮:** OIDC導入により、認証情報そのものはOkta側で管理され安全性が高まりますが、実装ミスにより**オープンリダイレクタやトークン漏洩**のリスクが生じないよう十分注意する必要があります。`state`パラメータの厳密な検証、`nonce`の利用、トークン受信後の適切な検証と保存/破棄処理など、Oktaのベストプラクティスに従った実装が必要です [12]。

以上より、**「不可能ではないが、SPIRAL固有の制約を踏まえた実装上の工夫が必要」**という結論です。現在の調査では明確な実装事例は見つかりませんでしたが、Googleアカウント連携のデモ実績 [16] もあることから、同様のアプローチでOkta連携も十分実現可能と考えています。不可能と判断すべき決定的な制約は見当たりませんでした。むしろ実装の成否は「SPIRAL側で許容されている機能の範囲内で、OIDCの手順を正確に実装できるか」にかかっています。

もし仮に実装を進める中で致命的な制約が判明した場合（例えば**SPIRAL環境からOktaへの通信がブロックされる**ようなことが万一あれば）、その段階で再度可否判断を見直すことになります。しかし現時点の情報では、そのようなブロッキング要因は見受けられませんでした。従って、**SPIRAL　ver.1でのOkta認証連携は「実現可能」という前提で詳細設計・検証を進めることが妥当**と結論づけます。

参考情報: 本調査で参照した資料や関連ドキュメントの抜粋を以下に示します。

- SPIRAL公式サポートサイト FAQ「スパイラルPHPで使用できない関数」より: 「スパイラルPHPでは、セキュリティ面に考慮しファイルシステムに影響を与える可能性のある関数・クラス等は使用できないようになっています。」 [1] 不可関数リストに`file_get_contents`や`fsockopen`など外部通信・ファイル操作系が含まれています [2] [3]。一方でcURL関連関数は`curl_file_create`以外禁止リストになく、実際に使用可能。
- スパイラル ナレッジサイト「WEBミーティングツールと連携して会議URLを自動発行するサンプル」より: 「APIのレート制限、トークン期限、ネットワーク制限（外部通信可否）にご注意ください。」 [4] 外部サービス(Google/Zoom)と連携するPHPコード内で`curl_init()`や`curl_exec()`によるHTTPS通信を実装している例があります [5]。
- Okta Integration NetworkのSpiral向けページより: SpiralはOktaとの公式統合でSAML認証に対応しており（OIDCは公式サポート外）、Okta経由のセキュアなアクセスを提供できる旨が紹介されています [13]。これは裏を返せば、SPIRALが標準機能としてOIDCプロトコルを備えていないことを意味し、OIDC連携はカスタム実装になることを示唆しています。
- SPIRALの`$SPIRAL->getSession()`についての開発者ドキュメントより: 「PHPの$_SESSIONをラップするクラスで、…セッション開始時にIDを再作成し、タイムアウト時にセッションをクリアする」と記載されており、内部でセッション管理が行われていることが読み取れます [6]。カスタムPHPでもセッションを利用した値の保存・取得が可能です [17] [18]。

以上を踏まえ、本件Okta認証導入の方向で引き続き詳細設計と検証を進めてまいります。

---

[1] スパイラルのPHP対応で、使用できない関数についての資料はありますか？ SPIRAL ver.1 サポートサイト
https://support.smp.ne.jp/faq/f0110/

2 3 7 8 10 11 スパイラルで使用できない関数とクラスの一覧（PHPバージョン：7.4） SPIRAL ver.1 サポートサイト
https://support.smp.ne.jp/manuals/dev/spiralphp9/

4 5 登録フォームからWEBミーティングツールと連携して会議URLを自動で発行するサンプルプログラム | 記事一覧 | SPIRAL ナレッジサイト
https://knowledge.spirers.jp/article/development/detail/15783

6 17 18 Docs For Class SpiralSession
https://beta.smp.ne.jp/help/phpdoc/spiral/SpiralSession.html

9 15 Docs For Class Spiral
https://beta.smp.ne.jp/help/phpdoc/spiral/Spiral.html

12 13 Spiral | Okta
https://www.okta.com/integrations/spiral/

14 スパイラル® 1.13バージョンアップのご案内
https://www.smp.ne.jp/update/ver113/

16 〖デモ〗Googleアカウントによるシングルサインオン（SSO） | SPIRAL ナレッジサイト
https://knowledge.spirers.jp/article/design/detail/8880