

【特許請求の範囲】**【請求項 1】**

デバイスによるリソースサーバーへのアクセスをユーザーが許可したことを示す認可コードを認可サーバーが発行するための認可コード要求を、前記認可サーバーに送信する送信手段と、

前記認可コード要求に対応する認可コード応答を前記認可サーバーから受信する受信手段と、

を有するデバイスであって、

前記デバイスが有するアプリケーションから、前記認可サーバーの所在するリージョンを特定する情報であるリージョン情報を前記デバイスが有するトークンプロバイダーが取得し、

取得されたリージョン情報に基づいて、前記認可コード要求を送信する送信先である認可エンドポイントを特定し、

前記送信手段は、

特定された認可エンドポイントに対して前記認可コード要求を送信し、

前記受信手段は、

前記送信手段によって送信された認可コード要求に対応する認可コード応答を受信するデバイス。

【請求項 2】

前記トークンプロバイダーは、

前記リージョン情報と前記認可エンドポイントとの紐付け情報を有し、

前記紐付け情報に基づいて、前記認可コード要求を送信するための前記認可エンドポイントを特定する請求項 1 に記載のデバイス。

【請求項 3】

前記デバイスが前記リソースサーバーにアクセスするための認可トークンを発行するトークン要求を、前記認可コード応答で受信した前記認可コードとともに前記認可サーバーに送信し、

送信されたトークン要求に応じて、前記認可トークンを前記認可サーバーから取得する請求項 2 に記載のデバイス。

【請求項 4】

前記トークンプロバイダーが有する前記リージョン情報と前記認可エンドポイントとの紐付け情報は、

前記トークン要求を送信する送信先であるトークンエンドポイントを含み、

前記トークンエンドポイントに対して前記認可コードとともに前記トークン要求を送信する請求項 3 に記載のデバイス。

【請求項 5】

前記トークンプロバイダーは、

前記ユーザーが許可したことによって、前記デバイスが前記リソースサーバーにアクセスできるように処理する認可フローが実行されたことにより、

前記アプリケーションにログインしているローカルユーザーを特定する情報であるローカルユーザー ID と、前記トークン要求で取得した認可トークンと、前記認可トークンを発行した認可サーバーの所在するリージョンを特定するリージョン情報との紐付け情報で管理し、

前記アプリケーションから受信した前記リージョン情報とローカルユーザー ID が、前記紐付け情報が有する前記リージョン情報とローカルユーザー ID と一致するかを判定する判定手段を有し、

前記判定手段によって一致することが確認された場合は、前記紐付け情報で特定される認可トークンを用いて、前記リソースサーバーが公開する Web サービスを前記デバイスが利用するためのリソース要求を前記リソースサーバーに送信し、

前記判定手段によって一致しないことが確認された場合は、前記リージョン情報と前記

10

20

30

40

50

認可エンドポイントとの紐付け情報と、前記アプリケーションから受信したリージョン情報で特定された認可エンドポイントとを用いて、前記認可サーバーに前記認可コード要求を送信する請求項 3 または 4 に記載のデバイス。

【請求項 6】

前記アプリケーションから受信したリージョン情報が前記紐付け情報に含まれていない場合、または前記アプリケーションからリージョン情報を取得できない場合、

デフォルトとして設定されたリージョン情報と前記紐付け情報とで特定された認可エンドポイントを用いて、前記認可サーバーに前記認可コード要求を送信する請求項 5 に記載のデバイス。

【請求項 7】

前記トークンプロバイダーが、

前記アプリケーションから受信するリージョン情報に依らずに予め設定されたリージョン情報である固定リージョン情報を有する場合、

前記アプリケーションから受信したリージョン情報と前記固定リージョン情報とが一致するかを判定し、

一致すると判定された場合は、前記固定リージョン情報と、前記リージョン情報と前記認可エンドポイントとの紐付け情報とに基づいて前記認可エンドポイントを取得し、取得された認可エンドポイントに対して前記認可コード要求を送信し、

一致しないと判定された場合は、前記アプリケーションに対してエラー情報を送信する請求項 1 乃至 6 のいずれか一つに記載のデバイス。

【請求項 8】

前記固定リージョン情報は、

前記デバイスの所在するリージョンを特定する国コードを前記デバイスの IP アドレスから取得し、取得された国コードに対して紐付いて管理されている請求項 7 に記載のデバイス。

【請求項 9】

前記トークンプロバイダーは、

前記固定リージョン情報を格納するテーブルである固定リージョン判定テーブルを有するかどうかを判定する第二の判定手段を有し、

前記第二の判定手段によって前記トークンプロバイダーが有すると判定された場合は、

前記固定リージョン判定テーブルに格納された固定リージョン情報に基づいて、前記認可エンドポイントを特定し、特定された認可エンドポイントに対して前記認可コード要求を送信し、

前記第二の判定手段によって前記トークンプロバイダーが有しないと判定された場合は、

前記アプリケーションから受信したリージョン情報に基づいて、前記認可エンドポイントを特定し、特定された認可エンドポイントに対して前記認可コード要求を送信する請求項 7 または 8 に記載のデバイス。

【請求項 10】

前記リージョン情報は、

前記ユーザーが前記デバイスにログインする際に指定したリージョン情報、または前記ユーザーが前記デバイスにおいて使用する言語に基づいて特定されたリージョン情報のいずれか一つである請求項 1 乃至請求項 9 のいずれか一つに記載のデバイス。

【請求項 11】

前記デバイスは、

前記デバイスに対してローカルユーザーがログインした際にログインコンテキストを生成し、

前記ログインコンテキストに前記リージョン情報が含まれているかどうかを判定し、

含まれていると判定された場合は、前記ログインコンテキストに含まれるリージョン情報に基づいて、前記認可コード要求を送信する送信先である認可エンドポイントを特定し

10

20

30

40

50

、

含まれていないと判定された場合は、前記ローカルユーザーに対して前記リージョン情報を要求するための入力画面を提供する請求項 1 乃至請求項 9 のいずれか一つに記載のデバイス。

【請求項 1 2】

コンピュータを、

デバイスによるリソースサーバーへのアクセスをユーザーが許可したことを示す認可コードを認可サーバーが発行するための認可コード要求を、前記認可サーバーに送信する送信手段と、

前記認可コード要求に対応する認可コード応答を前記認可サーバーから受信する受信手段として機能させる制御方法であって、 10

前記デバイスが有するアプリケーションから、前記認可サーバーの所在するリージョンを特定する情報であるリージョン情報を前記デバイスが有するトークンプロバイダーが取得し、

取得されたリージョン情報に基づいて、前記認可コード要求を送信する送信先である認可エンドポイントを特定し、

前記送信手段は、

特定された認可エンドポイントに対して前記認可コード要求を送信し、

前記受信手段は、

前記送信手段によって送信された認可コード要求に対応する認可コード応答を受信する制御方法。 20

【請求項 1 3】

コンピュータを、

デバイスによるリソースサーバーへのアクセスをユーザーが許可したことを示す認可コードを認可サーバーが発行するための認可コード要求を、前記認可サーバーに送信する送信手段と、

前記認可コード要求に対応する認可コード応答を前記認可サーバーから受信する受信手段として機能させるプログラムであって、

前記デバイスが有するアプリケーションから、前記認可サーバーの所在するリージョンを特定する情報であるリージョン情報を前記デバイスが有するトークンプロバイダーが取得し、 30

取得されたリージョン情報に基づいて、前記認可コード要求を送信する送信先である認可エンドポイントを特定し、

前記送信手段は、

特定された認可エンドポイントに対して前記認可コード要求を送信し、

前記受信手段は、

前記送信手段によって送信された認可コード要求に対応する認可コード応答を受信するプログラム。 40

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、認可サーバーの認可エンドポイントにアクセスするデバイスに関する。

【背景技術】

【0002】

MFPなどのデバイスが有するアプリケーションが、同じくデバイスが有するトークンプロバイダーを介して認可サーバーから認可トークンを取得する形態がある。認可トークンとは、認証されたユーザーの認可操作により権限移譲されたデバイスが、リソースサーバーが公開するAPIにアクセスすることを許可する事示すトークンである。取得した認可トークンを用いることで、ID、パスワード情報、認可情報などのユーザー情報を受 50

け渡すことなく、デバイスはリソースサーバーが公開するAPIを利用することができる。例えばデバイスがMFPであれば、リソースサーバーが提供するプリントサービスや帳票サービスなどのWebサービスを利用して、デバイスがデータの表示、印刷を実行することができる。

【0003】

認可トークンは、認可サーバーがOAuth 2.0と呼ばれる標準プロトコルの認可フローAuthorization Code Grantを実行することで発行される。具体的には、デバイスがリソースサーバーのWebサービスを利用することを、Webブラウザを介してユーザーが認可することで、認可サーバーはトークン要求を送信してきたトークンプロバイダーに対して認可トークンを発行する。トークン要求とは、トークンプロバイダーが認可サーバーから認可トークンを取得するためのリクエストである。

10

【0004】

また、その認可トークンをトークンプロバイダーが取得するために、トークンプロバイダーは認可サーバーに対して認可コード要求を送信する。認可コードとは、認可トークンを認可サーバーから取得するためのトークンであり、認可コード要求はその認可コードを取得するためのリクエストである。

【0005】

特許文献1には、デバイスが有するアプリケーションが同じくデバイスが有するトークンプロバイダーを介して認可サーバーから認可トークンを取得するシステムが開示されている。

20

【先行技術文献】

【特許文献】

【0006】

【特許文献1】特開2017-107396

【発明の概要】

【発明が解決しようとする課題】

【0007】

ユーザー情報はある特定の地域（以下、リージョン）の認可サーバーで管理されている。そのユーザー情報が複数である場合、それらのユーザー情報が同じリージョンの認可サーバーで管理されているとは限らない。具体的には、アプリケーションが複数のリージョンで配布されワールドワイドで使用されている場合、アプリケーションを使用するユーザーのユーザー情報はそのユーザーがいるリージョンの認可サーバーで管理される。そしてユーザー情報が所在する認可サーバーで認可トークンが発行され、発行された認可トークンを検証して、リソースサーバーがアプリケーションにWebサービスを提供する。ユーザー情報の所在に応じた認可サーバーが認可トークンを発行する理由は、個人情報保護や安全保障等の観点から別のリージョンへのユーザー情報の移動や共有に制限が設けられている場合があるからである。

30

【0008】

アプリケーションが複数のリージョンで利用することを想定する。その場合、そのアプリケーションはユーザー情報が所在するリージョンに関する情報（以下、リージョン情報）をトークンプロバイダーに送信する。トークンプロバイダーは受信したリージョン情報に基づいて、認可サーバーに対して認可コード要求を送信する。そのため、認可コード要求を送信する送信先である認可エンドポイントの選択肢が広がる。

40

【0009】

トークンプロバイダーは、アプリケーションにより指定されたリージョン情報から認可エンドポイントを特定する必要がある。認可エンドポイントとは、各リージョンに存在する認可サーバーに認可コード要求を送信する送信先を指定するアドレスである。

【0010】

本願発明は、アプリケーションから受信するリージョン情報に応じて、認可コード要求を送信するための適切な認可エンドポイントのURLをデバイスが特定することを目的と

50

する。

【課題を解決するための手段】

【0011】

デバイスによるリソースサーバーへのアクセスをユーザーが許可したことを示す認可コードを認可サーバーが発行するための認可コード要求を、前記認可サーバーに送信する送信手段と、前記認可コード要求に対応する認可コード応答を前記認可サーバーから受信する受信手段と、を有するデバイスであって、前記デバイスが有するアプリケーションから、前記認可サーバーの所在するリージョンを特定する情報であるリージョン情報を前記デバイスが有するトークンプロバイダーが取得し、取得されたリージョン情報に基づいて、前記認可コード要求を送信する送信先である認可エンドポイントを特定し、前記送信手段は、特定された認可エンドポイントに対して前記認可コード要求を送信し、前記受信手段は、前記送信手段によって送信された認可コード要求に対応する認可コード応答を受信する。

10

【発明の効果】

【0012】

本願発明を用いることで、アプリケーションから受信するリージョン情報に応じて、認可コード要求を送信するための適切な認可エンドポイントのURLをデバイスが特定することができる。

【図面の簡単な説明】

【0013】

20

【図1】本発明の実施形態に係る権限委譲システム。

【図2】各種装置のハードウェア構成図。

【図3】各種装置が有する機能。

【図4】OAuth 2.0のAuthorization Code Grantの処理フロー。

【図5】トークンプロバイダー440が有する機能。

【図6】トークンプロバイダー440の動作シーケンス。

【図7】実施例1における認可エンドポイント選択の動作を示したフローチャート。

【図8】実施例2における認可エンドポイント選択の動作を示したフローチャート。

【図9】Webブラウザ410、510におけるリージョンの入力画面の一例。

30

【図10】実施例1の処理、または実施例2の処理を実行する判定手段の一例。

【発明を実施するための形態】

【0014】

以下、本発明を実施するための最良の形態について図面を用いて説明する。尚、後述の認可サーバー200と認可サーバー201のどちらでもよい場合は「認可サーバー200、201」と記載し、リソースサーバー300とリソースサーバー301のどちらでもよい場合も「リソースサーバー300、301」と記載する。ただし、認可サーバー201が認可コードや認可トークンを発行する場合は、認可サーバー201で発行された認可トークンを用いてリソースサーバー301の公開するAPIがアクセスされるものとする。一方で、認可サーバー200が認可コードや認可トークンを発行する場合は、認可サーバー200で発行された認可トークンを用いてリソースサーバー300の公開するAPIがアクセスされるものとする。

40

【0015】

まず、図1を用いて、本発明の実施形態に係る権限移譲システムについて説明する。権限移譲システムは、図1に示すような構成のネットワーク上に実現される。WAN (Wide Area Network) 100はWWW (World Wide Web) システムによって構築されている。WAN 100と各種デバイス200～500はLAN (Local Area Network) 101を介して接続されている。

【0016】

認可サーバー200、201はOAuth 2.0を実現するためのサーバーであり、

50

認証要求の受信や認可コードの発行、管理等の処理を行う。図1では、認可サーバー200とリソースサーバー300、認可サーバー201とリソースサーバー301がLAN101で接続されている形態を示しているが、WAN100を介して接続される構成も可能である。また、認可サーバー200、201はLAN101を介して不図示のデータベースサーバーと接続されており、認可サーバー200、201が自身の機能を実現する際に用いるデータをそのデータベースサーバーに格納するように構成してもよい。図1では認可サーバー200とリソースサーバー300、認可サーバー201とリソースサーバー301をそれぞれ別のサーバーであるものとして説明しているが、同一のサーバー上に両サーバーの機能が構成されている形態でもよい。

【0017】

尚、リソースサーバー300と認可サーバー200（あるいはリソースサーバー301と認可サーバー201）との所在の形態は、同じリージョン内、あるいは同じシステム内である形態に限定されない。認可サーバー200で発行された認可トークンをリソースサーバー300が問い合わせることができれば、所在の形態は問わない。あるいは、リソースサーバー300が、認可トークンとともに受信した署名情報を検証する等の形態でもよい。

【0018】

デバイス400は例えばプリンターやMFP、もしくはPCやスマートフォン等が挙げられる。端末500は、ユーザーがWebブラウザ510を介して認可サーバー200、201に対するユーザーの認証要求やデバイス400に対するログイン操作等、各種デバイスの機能を利用することができる。端末500として具体的にはPCやスマートフォン等が挙げられる。

【0019】

デバイス400および端末500はそれぞれWebブラウザ410とWebブラウザ510を備える。ユーザーはWebブラウザ410またはWebブラウザ510を操作して後述の認可操作を実行する。デバイス400と端末500はLAN101を介して接続されている。以降、Webブラウザ410とWebブラウザ510のどちらで実行するかを問わない場合は、「Webブラウザ410、510」と称する。

【0020】

次に図2を用いて、認可サーバー200、201、リソースサーバー300、301およびデバイス400、端末500のハードウェア構成を説明する。なお、図2は一般的な情報処理装置のブロック図であり、本実施形態の各種デバイスには一般的な情報処理装置のハードウェア構成やIaaS（Infrastructure as a Service）として提供される情報処理装置の仮想的なハードウェア構成を適用できる。図2では、デバイス400を例に説明するが、リソースサーバー300、301や認可サーバー200、201、端末500も同様のハードウェア構成を有するものとする。

【0021】

CPU2001は、RAM2002、ROM2003、外部メモリ2011などからプログラムを取り出してプログラムの命令を実行し、デバイス400の制御を行うユニットである。後述のシーケンスはこのプログラムの命令が実行されることにより実現される。また、CPU2001はシステムバス2004に接続される各ブロックを制御する。

【0022】

RAM2002は、CPU2001が命令を実行する際に使用するワークメモリである。ROM2003や外部メモリ2011に保存されたOSやアプリケーション等のプログラムがRAM2002へとロードされ、そのプログラムの命令をCPU2001が順次読みだすことで命令を実行する。ROM2003は、アプリケーションプログラムおよびOSを含む組込済みプログラム、およびデータ等が記録されている記憶装置である。

【0023】

KBC（キーボードコントローラ）2005は、KB（キーボード）2009や不図示のポインティングデバイスからの入力を制御するユニットである。CRTC（Catho

10

20

30

40

50

de Ray Tube Controller) 2006は、CRTディスプレイ2010の表示を制御するユニットである。DKC (Disk Controller) 2007は外部メモリ2011に対するデータアクセスを制御するユニットである。NC (ネットワークコントローラ) 2008はWAN100やLAN101を介して接続された他のデバイスとの通信制御処理を実行する。なお、IaaSとして提供される仮想的な情報処理装置の場合は、KBC 2005やCRTC 2006等を備えず、NC 2008を介して接続される端末が備えるキーボードやCRTディスプレイから操作されるよう構成される。

【0024】

尚、後述の説明においては、特に断りのない限り、各種デバイスの機能が実行される際のハードウェアの主体はCPU 2001であり、ソフトウェアの主体はRAM 2002、ROM 2003、外部メモリ2011等にインストールされたプログラムであるものとする。

【0025】

次に図3を用いて、認可サーバー200、201、リソースサーバー300、301、デバイス400、端末500が有する機能について説明する。認可サーバー201は認可サーバー200と、リソースサーバー300はリソースサーバー301と同様の機能を有するため、図3の説明では認可サーバー200とリソースサーバー300を例に説明する。

【0026】

認可サーバー200は、認可サーバー部210とHTTPサーバー部220を有する。HTTPサーバー部220はWAN100を介してデバイス400と端末500に接続されており、Webブラウザ410、510や後述のアプリケーション420とHTTP通信を行う機能である。また、HTTPサーバー部220はSSL/TLSによる通信が可能であり、不図示の証明書ストアを有する。

【0027】

認可サーバー部210はHTTPサーバー部220を介してWebブラウザ410、510からの要求を受信し、受信した要求に対する結果を応答する機能である。具体的には、ユーザー認証の要求をHTTPサーバー部220がWebブラウザ410、510から受信し、認証が成功したユーザーのユーザー情報が紐づいた認証トークンを生成し、Webブラウザ410、510に認証トークンを通知する。認証トークンとは、ユーザーが認可サーバー200にログインしている事を示すためのトークン、またはユーザーが認可サーバー200において認証されているかを検証するためのトークンである。認証トークンを用いることで認可サーバー200はユーザーを識別することが可能となる。また、認可サーバー部210は、認可トークンに署名情報を付与するための秘密鍵を保持するように構成する事もできる。その場合は、この秘密鍵を用いて認可トークンに署名情報を付与し、署名情報付きの認可トークンをデバイス400に対して発行する。

【0028】

リソースサーバー300はリソースサーバー部310を有する。リソースサーバー部310は、Webサービスを提供するためのAPIを公開する機能である。なお、認可サーバー200と同様に、HTTPサーバー部を備え、HTTPサーバー部を介して外部との受送信を実行する形態でもよい。

【0029】

デバイス400はWebブラウザ410とアプリケーション420と認証部430とトークンプロバイダー440を有する。Webブラウザ410はWWWを利用するためのユーザーエージェントによって実現される機能である。Webブラウザ410はユーザーの操作により、認可サーバー200およびトークンプロバイダー440と通信を行う。端末500が備えるWebブラウザ510もWebブラウザ410と同様の機能である。アプリケーション420は、トークンプロバイダー440を介して認可サーバー200から認可トークンを取得する。取得した認可トークンを用いてリソースサーバー300

0 が公開する A P I を利用する。

【0030】

トークンプロバイダー440は、アプリケーション420からの認可トークン要求を受信し、認可サーバー200と通信を行う事で認可トークンを取得する。ユーザーはWebブラウザ410、510を用いて認可サーバー200およびトークンプロバイダー440と通信を行う事で認可操作を行う。

【0031】

ここで、認可トークン要求は、アプリケーション420が認可トークンを取得するためにトークンプロバイダー440に対して送信するリクエストである。一方のトークン要求は、トークンプロバイダー440が認可トークンを取得するために認可サーバー200、201に対して送信するリクエストである。このように、同じ認可トークンを取得するリクエストであっても、リクエストの受信元と送信先とが異なることでリクエストの呼称を変えていることを留意されたい。

【0032】

トークンプロバイダー440はベンダーデフォルトクレデンシャルとして、トークンプロバイダー440自身を証明するためにX.509形式で定められるクライアント証明書およびその秘密鍵を備える。そして、トークンプロバイダーが、クライアント証明書およびその秘密鍵を、認可サーバー200との通信を確立する際に利用する事で、認可サーバー200はトークンプロバイダーを認証することができる。

【0033】

認証部430はユーザーを認証するための機能である。ユーザーはデバイス400の機能を利用するために、デバイス400における不図示の入力画面においてローカルユーザーIDとローカルユーザーパスワードを入力する。入力情報を受信したデバイス400は、認証部430において予め登録されている情報（ローカルユーザーIDとローカルユーザーパスワード）と入力された情報とを照合することでユーザーの認証処理を行い、ログインコンテキストを生成する。なお認証処理の形態はこれに留まらず、例えば、ICカードを使った認証や指紋等の生体認証でもよい。

【0034】

ログインコンテキストは、デバイス400でローカルユーザーを識別するための情報であり、例えば、ローカルユーザーIDから構成される。ログインコンテキストは、デバイス400にローカルユーザーがログインした際にデバイス400のOS（不図示）において生成され、ログオフした際に消える。ローカルユーザーのログイン操作でログインコンテキストが生成されることにより、ログイン中のローカルユーザーがアクセス権限を有するWebページがWebブラウザ410上に公開される。ローカルユーザーのログオフ操作でログインコンテキストが消去されることにより、ローカルユーザーがアクセス権限を有するWebページを別のユーザーに公開することなく、セキュリティを担保することができる。

【0035】

このログインコンテキストはアプリケーション420と認証部430とトークンプロバイダー440で共有される。尚、ログインコンテキストにはローカルユーザー情報のみならず、ローカルユーザー情報に紐づくユーザー情報が所在するリージョン情報も含む形態でもよい。その場合、デバイス400においてログインコンテキストが生成された際に、アプリケーション420と認証部430とトークンプロバイダー440とでリージョン情報も共有される。共有されたリージョン情報とローカルユーザー情報とに基づいて、認可コード要求を送信する送信先の認可エンドポイントをトークンプロバイダー440が特定してもよい。

【0036】

本実施例ではデバイス400へのログイン処理についてユーザーが直接デバイス400を操作してログインする形態で説明するが、Webブラウザ510を介してリモートでログインする形態でもよい。その場合、認証部430は不図示のログイン画面をWebブ

10

20

30

40

50

ラウザー 510 に応答する。ユーザーはそのログイン画面にローカルユーザー ID とローカルユーザーパスワードを入力することでユーザーは認証される。その際、認証部 430 で生成されたログインコンテキストは、アプリケーション 420 と認証部 430 とトークンプロバイダー 440 で共有される。

【0037】

次に図 4 を用いてデバイス 400 が有するトークンプロバイダーが提供する OAuth 2.0 の処理について、認可サーバー 200、201 と、リソースサーバー 300、301、デバイス 400 と、Web ブラウザー 510 とを用いて説明する。尚、図 4 では認可サーバー 201 を例に説明するが、認可サーバー 200 の場合でも同様である。本実施例では、トークンプロバイダー 440 が提供する認証認可技術として OAuth 2.0 を例に説明するが、認可トークンを発行する形態であれば OAuth 2.0 以外の形態でもよい。また、図 4 の後述の説明において、デバイス 400 が主体となっている処理は、特に断りがない限りトークンプロバイダー 440 である。

【0038】

OAuth 2.0 を実行するための事前操作として、認可サーバー 201 にデバイス 400 を登録するための登録要求を行う (S0.0)。具体的には、デバイス 400 の登録要求は認可サーバー 201 の登録エンドポイント (図 4 ではエンドポイントを「EP」と記載) に対して送信され、デバイス 400 の起動時か、または後述の S1.1 の認可フローの開始時にデバイス 400 が未登録であった場合に開始される。登録要求の方法としては、デバイス 400 が能動的に認可サーバー 201 と通信する方法や、ユーザーが Web ブラウザー 510 を介して認可サーバー 201 にアクセスしてデバイス 400 を登録する方法が挙げられる。

【0039】

S0.0 の登録要求には、後述の認可確認画面に表示するためのデバイス名、説明、アイコン画像、および必須パラメーターであるリダイレクト URI が含まれる。リダイレクト URI とは、デバイス 400 が認可コード応答を送信する送信先を指定するアドレスである。認可コード応答については後述する。デバイス 400 の登録要求を受信した認可サーバー 201 はデバイス 400 を識別するためのデバイス ID と、デバイス 400 を認証するための秘匿情報であるデバイスシークレットとを発行し、デバイス 400 の登録応答としてデバイス 400 に送信する (S0.1)。認可サーバー 201 は、S0.1 でデバイス ID とデバイスシークレットと、S0.0 で受信した各種情報およびリダイレクト URI とを紐づけて保持する。デバイス 400 は、S0.1 で受信したデバイス ID とシークレットを保持する。以上が OAuth 2.0 を実行する事前操作であるデバイス 400 の登録フローである。

【0040】

次に、認可サーバー 201 においてユーザーを認証するためのフローを、図 4 を用いて説明する。ユーザーはデバイス 400 にログインする (S1.0)。デバイス 400 の認証部 430 は、ログインしたユーザーを特定するための情報であるログインコンテキストを生成し保持する。生成されたログインコンテキストからログインしたユーザーを特定する情報 (ローカルユーザー ID 等) を取得する事ができる。ユーザーが Web ブラウザー 510 を介して認可開始の URI にアクセスすることで、OAuth 2.0 の認可フローを開始する (S1.1)。デバイス 400 は認可フローを開始するための認可開始 URI にアクセスされると、認可サーバー 201 の認可エンドポイントに対して認可コード要求を送信する (S1.2)。認可コード要求には、デバイス ID、リダイレクト URI、state パラメーターを含む。

【0041】

state パラメーターは認可コード要求と認可コード応答とを一意に紐づけるための情報であり、CSRF (cross-site request forgery) 攻撃や、トークン置換 (以下、認可コード置換) 攻撃を防ぐために用いられる。そのため、state パラメーターは推測不可能でかつ重複しない値である必要がある。後述の認可コ

ード応答でデバイス400が受信するstateパラメーターとS1.2の認可コード要求で送信したstateパラメーターとの一致を検証し、さらに認可コード要求を実行したユーザーを識別するために、デバイス400で発行されるstateパラメーターはリダイレクトURIとログインコンテキストとが紐付いてデバイス400で管理される。

【0042】

S1.2で認可コード要求を受信した認可サーバー201は、ユーザーが認可サーバー201にログインしていなかった場合、Webブラウザ510にユーザーを認証するためのログイン画面を応答する(S1.3)。ユーザーは、Webブラウザ510を介してユーザーIDとパスワードを入力し、認可サーバー201に対して認証要求を行う(S1.4)。認証要求を受信した認可サーバー201は、S1.4で受信したユーザーIDとパスワードとの紐付け情報が事前に登録されている紐付け情報と一致するかを検証し、一致する場合は認証トークンを発行する。発行された認証トークンはWebブラウザ510のCookieに応答される。

【0043】

認可サーバー201はユーザーに対して、デバイス400の認可を同意するための認可確認画面をWebブラウザ510に応答する(S1.5)。ただし、S1.2で受信したデバイスIDとリダイレクトURIとの組み合わせが、認可サーバー201に予め登録されたデバイスIDとリダイレクトURIとの組み合わせと一致しない場合は、Webブラウザ510にエラー画面を応答する。これにより、不正なURIへのリダイレクト(転送)を防ぐことができる。また、認可サーバー201ログインしているユーザーが既に同一のデバイスIDで認可操作済みであった場合はS1.5を省略する事もできる。

【0044】

ユーザーによる認可操作(S1.6)の後、認可サーバー201は認可コードを発行し、デバイス400に対して認可コード応答として認可コードとstateパラメーターを送信する(S1.7)。具体的には、認可コードとstateパラメーターとをリダイレクトURIにクエリパラメーターとして付与し、リダイレクトURIで指定された宛先に認可コードとstateパラメーターとをリダイレクトするようにWebブラウザ510に送信する。S1.7で発行された認可コードは、認可サーバー201においてデバイスID、ユーザーID、リダイレクトURLと紐づけて保存される。

【0045】

リダイレクトURIに対して認可コード応答を受信したデバイス400は、認可コード応答に含まれるstateパラメーターと、デバイス400が管理するstateパラメーターとが一致するかを検証する。検証の結果、stateパラメーターが一致した場合、デバイス400は認可サーバー201のトークンエンドポイントに対して、デバイス400が認可トークンを取得するためのトークン要求を送信する(S2.0)。トークン要求にはデバイスID、シークレット、S1.7で取得した認可コード、および、S1.2で受信したリダイレクトURIが含まれる。

【0046】

S2.0でトークン要求を受信した認可サーバー201は、デバイスIDとシークレットの組み合わせが予め登録された組み合わせと一致するかを検証する。検証の結果、一致が確認されるとデバイス400が認証される。さらに認可サーバー201は、S2.0で受信した認可コードを保持しているか、保持している場合にはその認可コードは有効期限内か、認可トークンと紐づいたデバイスIDとリダイレクトURIがS2.0のトークン要求で受信したものと一致するか、を検証する。この検証により、S1.2の認可コード要求を送信したデバイス400とS2.0のトークン要求を送信したデバイス400とが一致するかを認可サーバー201で検証する事ができる。

【0047】

検証が成功した場合、認可サーバー201はデバイス400に対して認可トークンを発行し、トークン応答としてデバイス400に応答する(S2.1)。その際、認可トークンを再取得するためのリフレッシュトークンもデバイス400に対して発行し、トークン

10

20

30

40

50

応答として応答する事もできる。デバイス400は、S2.1で受信した認可トークンを用いてリソースサーバー301が公開するAPIにアクセスすることが可能となる。また、認可トークンを発行した後に、認可サーバー201で管理する認可コードを破棄することでリプレイ攻撃を防ぐことが可能となる。

【0048】

S2.1のトークン応答にリフレッシュトークンが含まれる場合、デバイス400においてログインコンテキストとリフレッシュトークンとが紐付けて管理される。それにより、次回以降のAPIへのアクセス時に認可操作(S1.2～S1.7)を実施せずに認可トークンを再度取得する事ができる。具体的には、S1.1の認可開始を受けた際に、デバイス400においてユーザーのログインコンテキストとリフレッシュトークンとが紐付いているかを確認する。紐づいていない場合は前述のOAuth2.0のフロー(S1.2以降の処理)を実施する。紐づいている場合は、認可サーバー201のトークンエンドポイントに対してリフレッシュ要求を行う(S2.2)。

【0049】

リフレッシュ要求はデバイスID、シークレット、およびリフレッシュトークンを含む。リフレッシュ要求を受信した認可サーバー201は、デバイスID、シークレットの組み合わせがS0.1で事前登録されたものと一致するかを検証する。一致が確認されデバイス400が認証されると、受信したリフレッシュトークンが認可サーバー201で保持されているか、保持されている場合はそのリフレッシュトークンは有効期限内か、さらにリフレッシュトークンに紐づいたデバイスIDがリフレッシュ要求のものと一致するかを検証する。これらの検証が全て成功した場合、認可サーバー201は認可トークンを発行し、デバイス400にトークン応答として認可トークンを送信する。その際、認可トークンを再取得するために新たなリフレッシュトークンを再発行し、トークン応答と同時にデバイス400に対して送信する形態も可能である。また、認可サーバー201において新たなリフレッシュトークンを発行した後に、認可サーバー201でそれまで管理されていたリフレッシュトークンを破棄する事でリプレイ攻撃を防ぐことができる。

【0050】

以上がOAuth2.0におけるAuthorization Code Grantの処理フローである。OAuth2.0による処理フローにより、認可サーバー201が管理するユーザー情報をデバイス400に送信する代わりに、認可サーバー201が認可トークンを発行し、デバイス400は発行された認可トークンを用いてリソースサーバー301が公開するAPIにアクセスすることができる。

【0051】

〔実施例1〕

ユーザー情報はある特定のリージョンの認可サーバーに登録されており、複数のユーザー情報が同じリージョンの認可サーバーで管理されているとは限らない。また、個人情報保護、安全保障等の観点から異なるリージョン間でのユーザー情報の移動や共有が制限される場合がある。

【0052】

その場合、トークンプロバイダー440は複数の認可サーバーの中からアプリケーション420が指定したリージョンに対応した認可サーバーを選択し、その認可サーバーに認可コード要求を送信する必要がある。実施例1では、ユーザー情報を管理する認可サーバーをトークンプロバイダー440が特定し選択する形態を説明する。

【0053】

まず、図5を用いて、トークンプロバイダー440が有する機能を説明する。トークンプロバイダー440は、エンドポイント選択部610、トークン取得部620、トークン管理部630、トークン配布部640を有する。

【0054】

エンドポイント選択部610は、OAuth2.0の認可フローにおいて認可サーバー200、201の認可エンドポイントに対する認可コード要求(図4のS1.2)を送

10

20

30

40

50

信する機能である。その際、エンドポイント選択部 610 は、エンドポイント選択部 610 が有するリージョン情報と認可エンドポイント URL の対応表を用いて、アプリケーション 420 から指定されたリージョンに応じた認可エンドポイント URL を選択する。エンドポイント選択部 610 が有する対応表の一例を表 1 に示す。また、本実施例では、認可サーバー 200 がリージョン「JP」に所在し、認可サーバー 201 がリージョン「EU」に所在するものとする。また、表 1 に認可エンドポイントが含まれるリージョン「GB」の認可サーバーは、図 1 では不図示である。

【0055】

【表 1】

表 1 エンドポイント URL データベース

No	リージョン	認可エンドポイント URL	トークンエンドポイント URL
1	JP	https://jp.example.com/oauth2/authorize	https://jp.example.com/oauth2/token
2	EU	https://eu.example.com/oauth2/authorize	https://eu.example.com/oauth2/token
3	GB	https://gb.example.com/oauth2/authorize	https://gb.example.com/oauth2/token

【0056】

表 1 は、「No.」（項目番号）、「リージョン」、「認可エンドポイント URL」、「トークンエンドポイント URL」のカラムを有し、リージョンを主キーとする。本実施例では「リージョン」として「JP」と「EU」が登録され、各リージョンに対して認可エンドポイント URL とトークンエンドポイント URL が関連付いて登録されている。表 1 へのデータ登録は外部のアプリケーション等から実行しても良く、登録の形態は問わない。

【0057】

トークン取得部 620 はトークンエンドポイントに対してトークン要求（S2.0）を送信する機能である。トークン取得部 620 は表 1 を参照して、アプリケーション 420 から指定されたリージョンに対応したトークンエンドポイントにトークン要求を送信する。

【0058】

トークン管理部 630 は、トークン取得部 620 が取得した認可トークンをローカルユーザー ID 毎に管理する機能である。トークン管理部 630 が管理するトークンデータベースの一例を表 2 に示す。

【0059】

10

20

30

40

【表 2】

表 2 トークンデータベース

No	ローカルユーザー ID	リージョン	認可トークン	リフレッシュトークン
1	Local__user1	JP	utbhhtpbtrmj uevnryy0enl qw9vaix	o23Tx1T1dLR iilmDq6CDDs
2	Local__user1	EU	t3geevy18cz kcb9lujmtgc hmuyivzg	ZTTcC8dNzTi F9FF84wF8wy
3	Local__user2	EU	t3geevy18cz kcb9lujmtgc hmuyivzg	ZTTcC8dNzTi F9FF84wF8wy

10

【0060】

表 2 は、「No」、「ローカルユーザー ID」、「リージョン」、「認可トークン」、「リフレッシュトークン」のカラムを有し、ローカルユーザー ID とリージョンを主キーとする。リフレッシュトークンは上記で説明した通り、OAuth 2.0 RFC 6749、6. Refreshing an Access Token にある、同じ認可フローを再度使用するための新しいアクセストークンである。表 2 のテーブルは、図 4 で示した認可フロー（S1.0～S2.2）が実行されたことで生成され、トークン管理部 630 が管理する。

20

【0061】

また、表 2 では、一つのローカルユーザー ID「Local__user1」に対して、複数のリージョン「JP」と「EU」が紐付いている。つまり、一つのローカルユーザー ID に対して紐付いている複数のユーザー情報がリージョン「JP」の認可サーバー 200 と、リージョン「EU」の認可サーバー 201 とに格納されていることになる。その場合、ユーザーがアプリケーション 420 においてリージョンを「JP」または「EU」に切り替えることで、認可トークンの要求先の認可サーバーを切り替えるものとするが、そのアプリケーション 420 の形態は問わない。

30

【0062】

トークン配布部 640 は、ローカルユーザー ID とリージョン情報とに基づいて認可トークンをアプリケーション 420 に送信する機能である。具体的には、アプリケーション 420 からリージョン情報とローカルユーザー ID を受信する。トークン配布部 640 は、受信したローカルユーザー ID とリージョン情報とに紐付く認可トークンを、表 2 に基づいてアプリケーション 420 に送信する。

40

【0063】

表 2 において、受信したローカルユーザー ID とアプリケーション 420 が指定したリージョン情報との組合せが存在しない場合、認可フロー（図 4）がまだ実行されていないものと見なされる。そして、エンドポイント選択部 610 は指定されたリージョン情報と表 1 とを用いてリージョンの認可エンドポイント URL を特定し、認可エンドポイント URL で特定される認可サーバー 200 から認可トークンとリフレッシュトークンとを取得し、ローカルユーザー ID とリージョン情報と共に、トークン管理部 630（表 2）に格納する。以上が、トークンプロバイダー 440 が有する機能である。

【0064】

次に図 6 を用いて、アプリケーション 420 がトークンプロバイダー 440 を介して、

50

認可サーバー 200 から認可トークンを取得し、取得した認可トークンを用いてリソースサーバー 300、301 が公開する API を実行するまでの処理を説明する。尚、図 6 の説明において、認可サーバー 200 とリソースサーバー 300 を用いて説明するが、認可サーバー 201 とリソースサーバー 301 とでも同様の処理が実行される。また、上記で説明済みの処理については、詳細な説明を省略する。

【0065】

まず、アプリケーション 420 はトークンプロバイダー 440 に対して認可トークン要求を送信する (S7. 1)。具体的には、アプリケーション 420 が認可トークン要求として、ローカルユーザー ID とリージョン情報とを送信する。アプリケーション 420 がリージョン情報を取得する形態はいくつか考えられる。例えば、デバイス 400 の Web ブラウザー 410 を介して、ユーザーからローカルユーザー ID とパスワードを入力する際に、ユーザーがリージョン情報を入力または選択する形態が考えられる。図 9 にリージョンを選択する画面の一例を示す。図 9 では、ローカルユーザー ID の入力欄として「ユーザー ID」、パスワードの入力欄として「パスワード」、リージョンをプルダウンで選択できる欄として「リージョン」が設けられている。

10

【0066】

または、ユーザーによるデバイス 400 へのログイン操作の際に生成されたログインコンテキストに、リージョン情報が含まれているかどうかをアプリケーション 420 が判定する形態も考えられる。含まれていると判定された場合は、上記でも説明した通り、ログインコンテキストに含まれるリージョン情報に基づいて、認可エンドポイントを特定する。含まれていないと判定された場合は、Web ブラウザー 410、510 を介して、ユーザーにリージョンを入力または選択させる画面を提供する。

20

【0067】

それ以外の形態で、予めアプリケーション 420 または Web ブラウザー 410 等に設定されたユーザーの使用言語の情報に基づいて、特定されたリージョン情報をアプリケーション 420 がトークンプロバイダー 440 に送信する形態等も考えられる。

【0068】

トークンプロバイダー 440 は S7. 1 で受信したローカルユーザー ID とリージョン情報から、トークンプロバイダー 440 のトークン管理部 630 が有するトークンデータベース (表 2) を用いて、受信したローカルユーザー ID とリージョン情報との紐付け情報と一致するレコードを検索する。ローカルユーザー ID とリージョン情報との紐付け情報と合致するレコードが存在した場合、ローカルユーザー ID とリージョン情報とに紐づく認可トークンをアプリケーション 420 に送信する (S7. 6)。

30

【0069】

一致するレコードが存在しなかった場合、トークンプロバイダー 440 は、認可トークン要求で受信したリージョン情報と表 1 から認可エンドポイント URL を特定する。特定された認可エンドポイント URL を用いて図 4 の S1. 3 ~ S2. 1 で示した処理を実行して認可サーバー 200 から認可トークンを取得する。

【0070】

S7. 1 で認可トークン要求を受信したトークンプロバイダー 440 は、認可サーバー部 210 に認可コード要求を送信する (S7. 2)。S7. 2 は図 1 の S1. 2 と同様の処理であるため、詳細な説明は省略する。認可コード要求を受信した認可サーバー部 210 は、図 1 の S1. 3 ~ S1. 6 の処理を実行した後、認可コードをトークンプロバイダー 440 に送信する (S7. 3)。S7. 3 は図 1 の S1. 7 と同様の処理であるため、詳細な説明は省略する。

40

【0071】

S7. 3 で認可コード応答を受信したトークンプロバイダー 440 は、認可サーバー 200 のトークンエンドポイントにトークン要求を送信する (S7. 4)。S7. 4 の処理は図 1 の S2. 0 の処理に相当するので詳細な説明は省略する。トークン要求を受信した認可サーバー部 210 は、トークン応答として認可トークンとリフレッシュトークンとを

50

トークンプロバイダー 440 に送信する (S7. 5)。S7. 5 の処理は図 1 の S2. 1 の処理に相当するので詳細な説明は省略する。

【0072】

S7. 5 の実行後、トークンプロバイダー 440 は、S7. 5 で取得した認可トークンとリフレッシュトークンと、S7. 1 でアプリケーション 420 から取得したローカルユーザー ID とリージョン情報とを、トークンデータベース (表 2) に登録する。

【0073】

S7. 5 でトークン応答を受信した後、トークン配布部 640 はアプリケーション 420 に対し、S7. 5 で取得した認可トークンを送信する (S7. 6)。認可トークンを取得したアプリケーション 420 は、認可トークンを用いてリソースサーバー部 310 に対してリソース要求を送信する (S7. 7)。以上が、アプリケーション 420 がトークンプロバイダー 440 を用いて、認可サーバー 200 から認可トークンを取得し、取得した認可トークンを用いてリソースサーバー 300、301 が公開する API を実行するまでの処理である。

【0074】

次に、図 7 を用いて、トークンプロバイダー 440 が認可トークンを取得する際の認可エンドポイントの選択処理を説明する。

【0075】

まず、トークン配布部 640 は、アプリケーション 420 から認可トークン要求としてローカルユーザー ID とリージョン情報を受信する (S8. 1)。今回は、ローカルユーザー ID 「Local_user2」とリージョン情報「JP」を認可トークン要求としてトークン配布部 640 が受信したものとして説明する。

【0076】

トークン管理部 630 はトークンデータベース (表 2) を用いて、受信したローカルユーザー ID と一致するレコードがないかを確認する (S8. 2)。表 2 より、予め保存されているレコードと、ローカルユーザー ID とリージョン情報との紐付け情報が一致しないことがわかる。一致するレコードがあることが確認されると、そのレコードに含まれる認可トークンをアプリケーション 420 に送信する (S8. 10)。

【0077】

レコードがないことが確認されると、エンドポイント選択部 610 は、S8. 1 で指定されたリージョンに基づいて、認可エンドポイント URL を特定するための判定を行う (S8. 3)。具体的には、エンドポイント選択部 610 はエンドポイント URL データベース (表 1) に基づいて、指定されたリージョン「JP」と合致する認可エンドポイントを取得し設定する (S8. 4)。今回、認可エンドポイント URL として取得した URL は、表 1 より「https://jp.example.com/oauth2/authorize」である。S8. 1 の認可トークン要求で受信するリージョン情報に依じて、S8. 4～S8. 7 のいずれかの処理が実行される。アプリケーション 420 によってリージョンが指定されなかった場合、あるいは指定されたリージョンが表 1 に含まれていない場合はデフォルトとしてリージョンが「GB」であるものとして S8. 7 の処理を実行する。ただし、リージョン「GB」に所在する認可サーバーは、認可サーバー 200、201 以外の認可サーバー (図 1 では不図示) のことである。

【0078】

トークン取得部 620 は、S8. 4～S8. 7 で特定された認可エンドポイント URL に対して認可コード要求を送信する (S8. 8)。S8. 8 は、図 4 の S1. 2 (あるいは図 6 の S7. 2) に相当する。S8. 8 の処理の結果、取得した認可コードを用いて、認可サーバー 200 にトークン要求を送信する (S8. 9)。S8. 9 は、図 4 の S2. 0 (あるいは図 6 の S7. 4) に相当する。S8. 9 の処理の結果、取得した認可トークンをアプリケーション 420 に送信する (S8. 10)。以上が、トークンプロバイダー 440 が認可トークンを取得する際の認可エンドポイントの選択処理である。

【0079】

10

20

30

40

50

本実施例により、アプリケーション４２０からリージョンを指定された際に、トークンプロバイダー４４０が適切な認可エンドポイントを選択し、指定されたリージョン情報に応じた認可サーバーに対して認可コード要求を送信することができる。

【００８０】

また、本実施例ではアプリケーション４２０からローカルユーザーＩＤを取得する形態で説明したが、Ｓ１．０（図４）でユーザーがデバイス４００にログインした際に認証部４３０で生成されたログインコンテキストから、ローカルユーザーＩＤを特定する形態でもよい。

【００８１】

〔実施例２〕

実施例１では、個人情報保護、安全保障等の観点から異なる間でのユーザー情報の移動や共有が制限される場合を考慮して、アプリケーション４２０がトークンプロバイダー４４０に対してリージョンを指定する形態を示した。しかし、アプリケーション４２０がリージョンを指定する形態以外にも、予めトークンプロバイダー４４０において固定のリージョンが設定されている形態もある。

【００８２】

実施例２では、認可コードの取得先であるリージョンが固定で設定されているか否かの判定結果に応じて、固定のリージョン情報で特定される認可サーバーに認可コード要求を送信するか、または実施例１の形態を実行するかに切り替える形態を説明する。尚、実施例２で説明を省略した部分は実施例１と同様である。

【００８３】

まず、図８を用いて、トークンプロバイダー４４０が認可サーバーに認可コード要求を送信する処理を説明する。トークン配布部６４０は、アプリケーション４２０から認可トークン要求としてリージョン情報を受信する（Ｓ９．１）。

【００８４】

エンドポイント選択部６１０は、Ｓ９．１で受信したリージョン情報は使用せず、デバイス４００の設置場所に基づいて認可エンドポイントＵＲＬを特定する。具体的にトークンプロバイダー４４０は、デバイス４００のＮＣ２００８に設定されたＩＰアドレスから国コード（ＩＣＡＮＮなどで使用されているｃｃＴＬＤにおける国コード）を取得する（Ｓ９．２）。国コードの取得は、エンドポイント選択部６１０がＩＰアドレスから国コードを特定できるｍａｘｍｉｎｄ社の提供するＧｅｏＩＰデータベースを検索することで実行される。ＧｅｏＩＰデータベースを検索する形態以外にも、デバイス４００のロケール情報に設定された国情報を設置場所の国コードとして取得しても良い。今回は、デバイス４００の設置場所から国コードが「ＦＲ」であるものとする。

【００８５】

エンドポイント選択部６１０はＳ９．２で取得した国コードが、トークンプロバイダー４４０に設定されているリージョン情報と一致するかを判定する（Ｓ９．３）。エンドポイント選択部６１０が有する固定リージョン判定テーブルに基づいて実行される。固定リージョン判定テーブルの一例を表３に示す。

【００８６】

【表３】

表３ 固定リージョン判定テーブル

Ｎｏ．	固定リージョン	国コード
１	ＪＰ	ＪＰ
２	ＥＵ	ＦＲ， ＩＴ， ＤＥ， ＥＳ

【００８７】

表３は、「Ｎｏ．」（項目番号）、「固定リージョン」、「国コード」（ＩＣＡＮＮな

10

20

30

40

50

どで使用されているccTLDにおける国コード)のカラムを有するテーブルである。「固定リージョン」は、「国コード」に対してトークンプロバイダー440が設定すべきリージョンのことである。エンドポイント選択部610は、S9.2で取得した国コードに対して固定リージョンが存在するかを、固定リージョン判定テーブル(表3)より判定する。存在しない場合はS9.8に進み、アプリケーション420から指定されたリージョンから認可エンドポイントURLを取得し設定する(S9.9~S9.12)。S9.8とそれ以降の処理(S9.9~S9.12)は、実施例1の処理内容(図7のS8.3~S8.7)と同様なので、詳細な説明は省略する。

【0088】

本実施例では、S9.2で取得した国コードに対して、表3の固定リージョン判定テーブルに固定リージョン情報が存在するか否かで実施例1の処理に切り替えるか否かを判定したが、その判定方法には限定されない。例えば、トークン配布部640がアプリケーション420から認可トークン要求を受信した後に、エンドポイント選択部610が固定リージョン判定テーブルを有するかどうかを判定(図10のS10.1)する形態も考えられる。この判定手段は、S8.1(またはS9.1)の処理より前に実行されても良い。S10.1で表3が存在すると判定された場合は、実施例2(図8)のS9.2以降の処理に進む。S10.1で表3が存在しないと判定された場合は、実施例1(図8)のS8.2以降の処理に進む。

【0089】

S9.3で取得した国コードに対して固定リージョンが存在することを表3で確認された場合、固定リージョン情報とアプリケーション420から受信したリージョン情報とを比較する(S9.4)。比較して一致が確認された場合はS9.6に進み、一致が確認できなかった場合はS9.5に進む。今回は、S9.2で取得した国コードが「FR」なので、表3よりその固定リージョン「EU」が存在することがわかる。S9.1で取得したリージョン情報が「FR」であればS9.6に進む。

【0090】

一方、S9.1で取得したリージョン情報が「CN」であれば、トークンプロバイダー440の固定リージョン「FR」と一致しないため、S9.5に進む。具体的には、トークンプロバイダー440はアプリケーション420に対してエラー情報を送信する(S9.5)。エラー情報には、トークンプロバイダー440に対して固定で設定された固定リージョン情報と、取得した国コードに基づいて特定されたリージョン情報が一致しなかった旨を含めてもいいが、エラー情報の形態は特に問わない。

【0091】

S9.4で一致すると判定された場合、エンドポイント選択部610は取得した国コードまたは固定リージョン情報と、エンドポイントURLデータベース(表1)とを用いて、認可エンドポイントを特定する(S9.6)。

【0092】

S9.6またはS9.9~S9.12で認可エンドポイントURLが特定された後、トークン取得部620は、特定された認可エンドポイントURLに対して認可コード要求を送信する(S9.7)。

【0093】

以上が、トークンプロバイダー440が認可トークンを取得する際の認可エンドポイントを固定する処理である。

【0094】

[その他の実施例]

上記の実施例において、リージョン情報が「JP」と「EU」(デフォルトのリージョン情報を「GB」)であるものとして説明したが、リージョン情報の種類やその数は上記で挙げた例に限定されない。

【0095】

また、上記のフローにおいて、アプリケーション420によってリージョンが指定され

10

20

30

40

50

なかった場合、あるいは指定されたリージョンが表 1 に含まれていない場合は、デフォルトとして設定されているリージョン情報を用いて処理を進める形態を説明したが、エラー情報を送信して処理を終了する形態でもよい。

【0096】

上述の実施形態の 1 以上の機能を実現するプログラムを、ネットワーク又は記憶媒体を介してシステム又は装置に供給し、そのシステム又は装置のコンピュータにおける 1 つ以上のプロセッサがプログラムを読み出し実行する処理でも実現可能である。また、1 以上の機能を実現する回路（例えば、ASIC）によっても実現可能である。

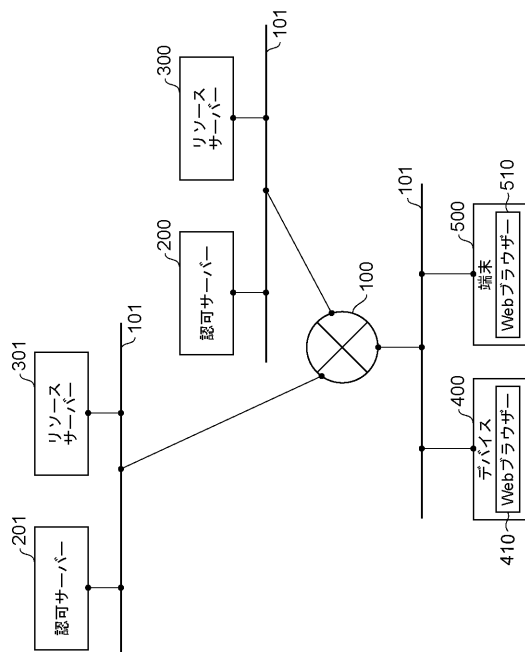
【符号の説明】

【0097】

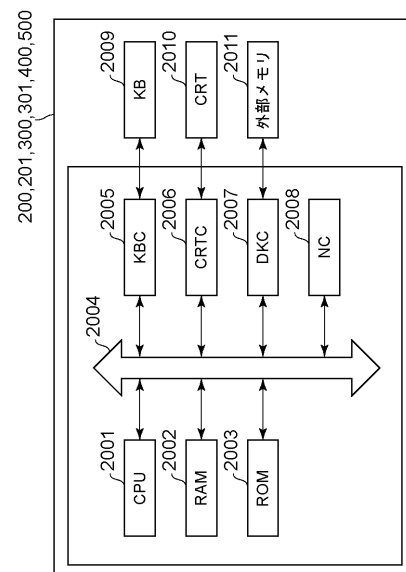
- 200、201 認可サーバー
- 420 アプリケーション
- 440 トークンプロバイダー

10

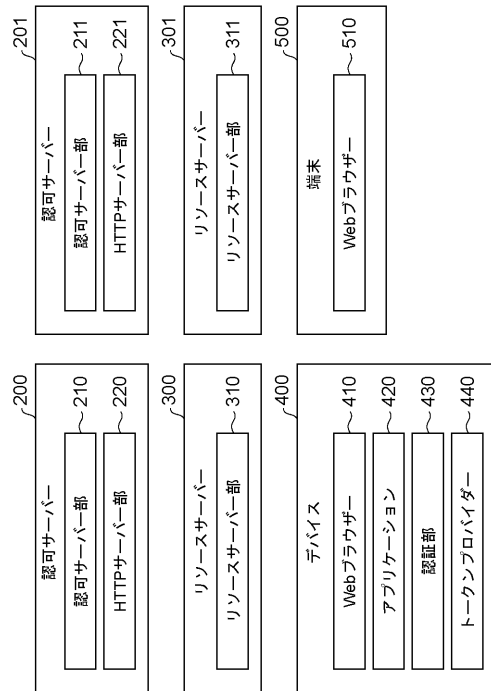
【図 1】



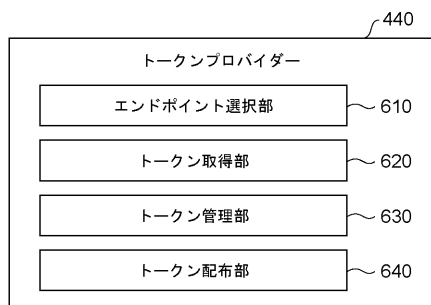
【図 2】



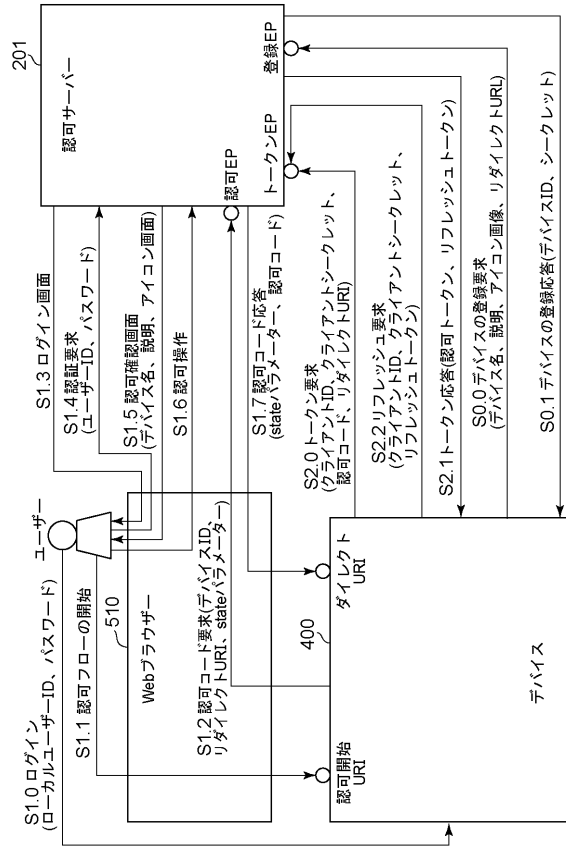
【図 3】



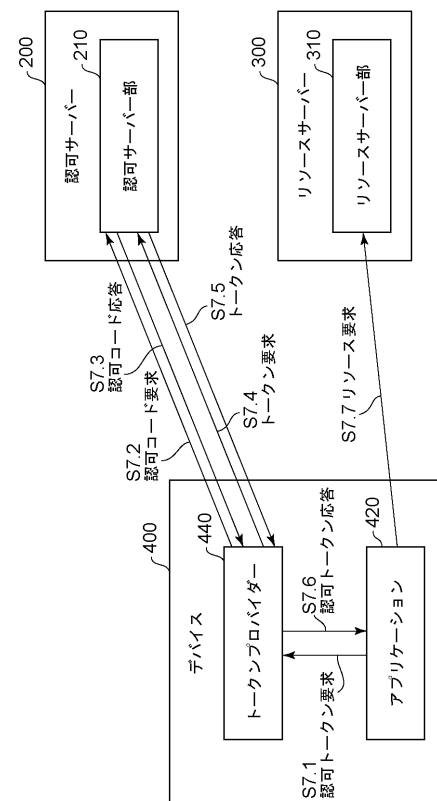
【図 5】



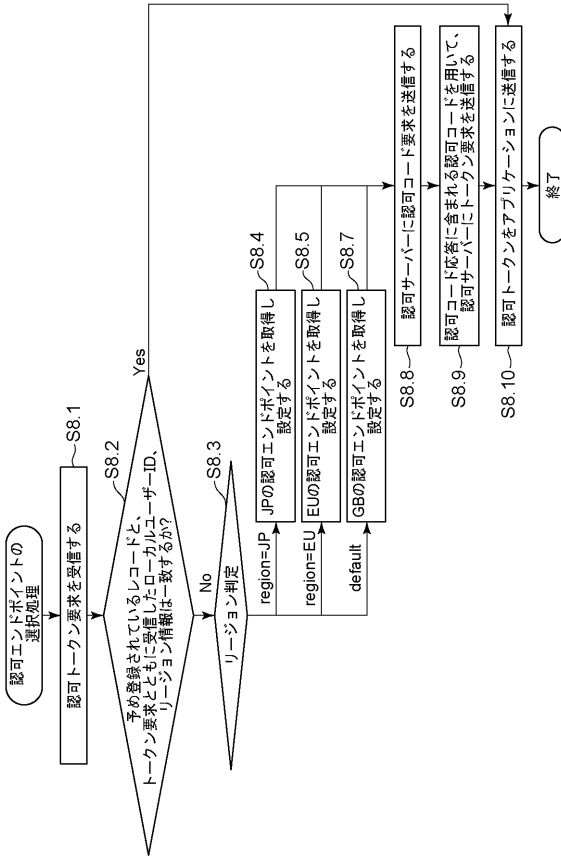
【図 4】



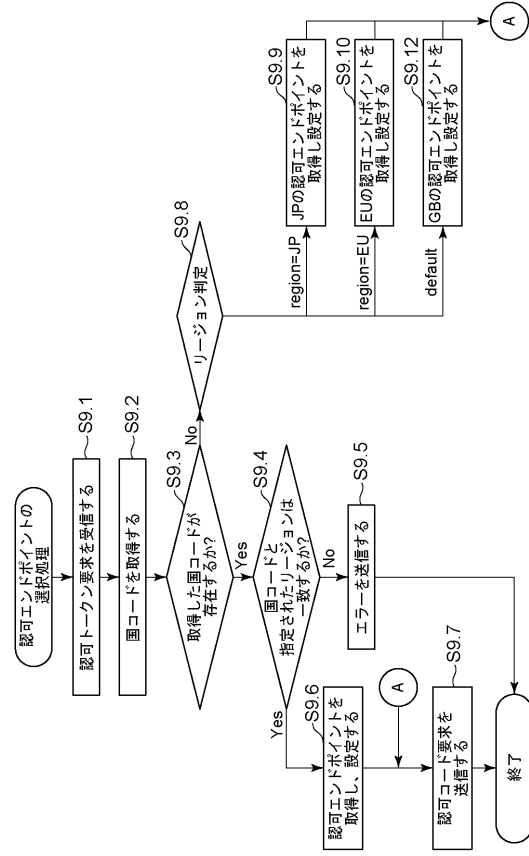
【図 6】



【図 7】



【図 8】



【図 9】

ユーザーID	
パスワード	
リージョン	Default ▼ Default JP EU

【図 10】

