OAuthへのコンシューマ認可の組み込みに関する研究

江波戸　太基[†]　松本　茂也[†]　友野　敬大[††]　上原　稔[†††]　島田　裕次[†††]
東洋大学工学部情報工学科[†], 東洋大学工学研究科情報システム専攻[††],
東洋大学総合情報学部総合情報学科[†††]

1 はじめに

近年、ITの飛躍的な進歩によってITが企業や家庭で広く使われるようになった。一方で、個人情報や技術情報などの漏洩や売買、財務および決算報告書等の改ざんなども頻発しており、情報漏洩や虚偽記載を防ぐために内部統制の必要性が高まってきている。また、SaaS等、複数のWebサービスをマッシュアップさせたサービスも注目され始めている。しかし、マッシュアップサービスでは、個人のクレデンシャル（IDおよびパスワード）を全て渡してしまう問題がある。これは、企業内部の統合システムにおいてもいえる問題であり、個々のサービスを利用する度にクレデンシャルを全て渡して認証を行うマッシュアップサービスでは個人情報流出の恐れがあり、セキュリティ上、適切ではない。

マッシュアップサービスにおいて、一方のサービスに登録してある個人のリソースに他方のサービスがその個人のリソースへクレデンシャルを渡さずにアクセスするためのプロトコルとして、OAuth[2]がある。既存のOAuthの仕様ではコンシューマからのリクエストの回数に制限がないので、ユーザがリクエストを拒否し続けているにも関わらずコンシューマは何度でもリクエストを送ることができてしまう脆弱性がある。

そこで本研究ではコンシューマに対してリクエストの回数を制限するOAuthを提案し、実装したのち、脆弱性に対する有効性の検証および評価を行う。

2 関連研究

2.1 OAuth

OAuthとはユーザ、コンシューマ、サービスプロバイダの三者間でセキュアな通信を行うプロトコルである。OAuthではトークンと呼ばれる一意のキーのやりとりで通信を行う（図1参照）。

ユーザは、コンシューマに対しサービス利用のリクエストを要求する（図1のa）。コンシューマは、サービスプロバイダに対しリクエストトークンを取得する（同b）。コンシューマは、ユーザをサービスプロバイダにリダイレクトさせ、取得したリクエストトークンの認可を促す（同c）。認可が完了すると、ユーザはコンシューマに戻り、アクセストークンの取得を要求する（同d）。ユーザは、取得したアクセストークンを用いてサービスを利用する（同e）。以上のフローにより、ユーザはコンシューマにIDとパスワードを渡すことなく、ユーザのリソースを利用するコンシューマのサービスを利用できるようになる。

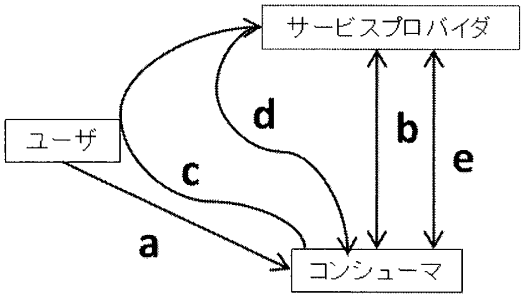

図1　OAuthのフロー

3 OAuthへのコンシューマ認可の組み込みの提案

本研究では、コンシューマ認可を行うOAuthの開発と、そのプロトコルを実装するためのシステムの構築という2つのアプローチで研究を進める。

3.1 コンシューマ認可を行うOAuthの開発

コンシューマ認可を行うOAuthは、一度拒否されたコンシューマは一定回数の拒否がなされた場合、それ以降のOAuthを受け付けないという仕組みである。ユーザによるトークンの拒否が一定回数以上行われたコンシューマは、そのコンシューマキーがサービスプロバイダ側で管理するリスト（blacklist）に格納される。blacklistの対象となったコンシューマは、ユーザがリダイレクトされる過程（図1のc）でblacklistの対象となっていることをブラウザの標準出力により伝える。ユーザの承認があればblacklistから取り出される。これらの一連の動作は、図2のとおりである。図2の“deny_count”とはユーザがトークンを拒否する回数であり、“n”とはblacklistに入れられるまでの拒否回数である。

3.2 プロトコルを実装するためのシステム構築

The Study on Implimentation Authorizing Consumer for OAuth.
Taiki Ebato, Shigeya Matsumoto, Akihiro Tomono, Minoru Uehara, Yuji Shimada
[†]Dept. of Information and Computer Sciences, Toyo Univ.
[††]Dept. of Open Information Systems, Toyo Univ.
[†††]Dept. of Information Sciences and Arts, Toyo Univ.

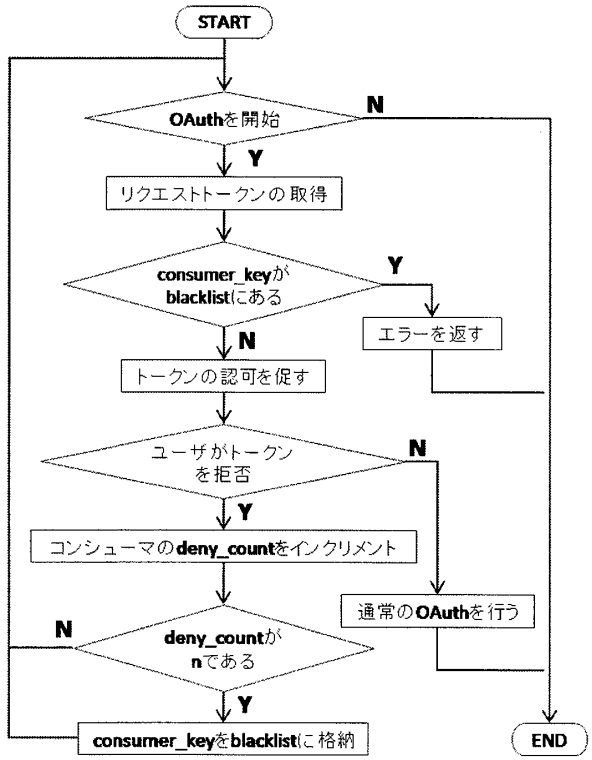

図2　コンシューマ認可のフローチャート

コンシューマ認可を行うOAuthを実装するためのシステムは学内における履修管理システムを想定する。ユーザとして学生、サービスプロバイダとして住所管理のデータベース、コンシューマとして履修管理のデータベースを用意する。アクセストークン取得後は、コンシューマの履修管理システムから検索したい学生の成績および氏名を取得し、サービスプロバイダの住所管理システムからその学生の住所を検索し、表示するアプリケーションである。

4 実装

4.1 コンシューマの実装

コンシューマでは、学生の氏名およびその学生の成績を管理するデータベースをMySQLで作成した。アプリケーションの概要は、ユーザがブラウザで学生の氏名を入力し、その氏名と一致する住所をサービスプロバイダから取得するというものである。アプリケーションおよびOAuthの機能を実装するために使用したライブラリの言語はPHPであり、主にGoogle Codeのライブラリ[3]を用いた。

4.2 サービスプロバイダの実装

サービスプロバイダでは、コンシューマから渡されたアクセストークンの判定処理のほかに、学生の氏名および住所、blacklistを管理するデータベースをSQLite3で作成した。アプリケーションの概要は、コンシューマから送られた学生の氏名に一致する住所をコンシューマに返すというものである。アプリケーションおよびOAuthの機能を実装するために使用したライブラリの言語はRubyであり、サンプルプログラム[1]に機能を追加することで実装した。

本研究ではblacklistの対象となっているコンシューマにその旨を通知する機能を実装しなかった。これは、コンシューマがブラウザ上で視覚的にOAuthのフローを行わず、サービスプロバイダの仕様に依存するからである。また、同様の理由により、blacklistの対象であるコンシューマがユーザの承認によりblacklistから取り出される機能を実装しなかった。これをOAuthの一連のフローで実装しなかった理由は、ユーザが誤ってblacklistの対象であるコンシューマに承認を与えてしまう可能性を考慮したものである。即ち、この機能もサービスプロバイダに依存する。

5 評価

実験は、コンシューマ、サービスプロバイダ共にUbuntu上で起動させ、本研究で実装したOAuthの機能およびアプリケーションの動作確認を行った。2つのコンシューマを用意し、コンシューマ認可機能の動作確認と、アクセストークンを取得する機能（通常のOAuthの機能）の動作確認をそれぞれ行った。アプリケーションの動作確認は、アクセストークンを取得したコンシューマを用いて動作確認を行った。

コンシューマ認可機能の動作確認では、nを3に設定し実験を行い、トークンを3回拒否されたコンシューマは、以降のトークンの承認が自動的に拒否され、期待通りの結果が得られた。nを3に設定した理由は、1、2度目はユーザが誤って拒否をクリックしてしまう可能性を考慮したものである。

アプリケーションの動作確認では、アクセストークンの判定処理およびコンシューマとサービスプロバイダ間のデータのやり取りが正常に行われ、期待通りの結果が得られた。

6 今後の課題

nの回数による有効性や、blacklistの対象となっているコンシューマにその旨を通知する方法、ユーザの承認があればblacklistから取り出される機能の実装方法を提案、検討する必要がある。また、アプリケーションの機能は実用性には及ばず、より厳密なパラメータの送受信方法および機能の充実を追求し、作成する必要がある。

参考文献

[1] ゼロから学ぶOAuth
http://gihyo.jp/dev/feature/01/oauth
[2] OAuth Core 1.0 Revision A 日本語訳
http://tzmtk.pbworks.com/OAuthCore10aJP
[3] oauth-php - Project Hosting on Google Code
http://oauth.googlecode.com/svn/code/php/