OAuth/OpenIDConnect実装におけるセキュリティ状況の調査

菊田 翼[2,a)]　齋藤 孝道[1]　小芝 力太[2]

概要：Webサイトを利用する際の認証において，ソーシャルメディアのアカウントを利用するソーシャルログインと呼ばれる仕組みがある．ソーシャルログインはOAuthやOpenIDConnectにより実装されることがある．しかし，Webサイト作成側での実装によっては，プライバシー上の問題を引き起こすことや攻撃に対し脆弱となることが知られている．本論文では，Webサイトのログインページの認証フローを辿ることで，そのサイトにおけるソーシャルログインの実装状況を調査した．その結果，SNSからのアクセス権限を必要以上に取得しているWebサイト，実装上の欠陥により脆弱性が残っている可能性を持つWebサイトが500サイト中28サイトで確認された．

Tsubasa Kikuta[2,a)]　Takamichi Saito[1]　Rikita Koshiba[2]

Abstract: When a website authenticates the users, it is applied to so-called social login in EC site. The social login is known to use a social media account, such as Facebook, Google, and Twitter. In the case, a website is applied to the use of OAuth and OpenIDConnect. However, the implementation of the website may be caused privacy concerns or be vulnerable to attacks. In this paper, by crawling the login pages of 500 Japanese EC sites and tracing the authentication flows, we investigated the implementation status of social login and their security against CSRF. As a result, we observed 28 websites that acquired users' permissions from SNS more than necessary, and some sites were vulnerable due to improper implementation.

1. はじめに

Webアプリケーションを利用する際に，GoogleやFacebookといったソーシャルメディアのアカウントを利用した認証方法が存在する．本論文ではこの認証方法をソーシャルログインと呼ぶ．ソーシャルログインの実装には，OAuthやOpenIDConnectなどの認可フレームワークが用いられていることが多い．また，OAuthの実装では，OAuth1.0[1]とOAuth2.0[2][3]が利用される．しかし，Webアプリケーション構築側でのソーシャルログインの実装により，プライバシー上の問題を引き起こすことや攻撃に対し脆弱となること[4]が知られている．

本論文では，Webサイトのログインページの認証フローを辿り，対象のWebサイトにおけるソーシャルログインの実装状況を以下の3点に注目して調査した．

- 国内の主要なWebサイトのうち，Google，Facebookおよび Twitterを利用したソーシャルログインの実装数
- ソーシャルログインを行う際に，クライアント（Webアプリケーション）が利用者に要求しているアクセス権限
- クロスサイトリクエストフォージェリ（CSRF）脆弱性を持つ可能性のある状態で運用しているクライアント数

本論文では，Top Sites in Japan - Alexa[5]における日本の上位500サイト（以降，上位500サイトと呼ぶ）を調査対象とした．

調査の結果，以下のことが確認できた．

- 上位500サイト中，重複も含め，ソーシャルログインとして，Google利用が58件，Facebook利用が56件，Twitter利用が27件存在した
- ソーシャルログインが実装されていたクライアントの中で，必要以上にアクセス権を要求しているものが14件存在した

[1] 明治大学
Meiji University
[2] 明治大学大学院
Graduate School of Meiji University
[a)] ce195013@meiji.ac.jp

- CSRF 脆弱性を持つ可能性のある状態で運用しているクライアントが上位 500 サイト中 29 件存在した

2. 関連知識

2.1 OAuth

OAuth とは，利用者に代わり，リソースにアクセスするための方法を，クライアントに提供する認可フレームワークである．また，利用者はクライアントにユーザ ID やパスワードを共有することなく，リソースへの第三者アクセスを認可できる．また，RFC6749[2] によると，推奨されている最新バージョンは OAuth2.0 である．

ここで，図 1 に OAuth2.0 におけるアクセス権限委譲のフローを示す．また，OAuth2.0 におけるアクセス権限委譲のフローでは，以下の 3 つの主体が存在する．

- 利用者
  リソースサーバのリソースオーナーである
- リソースサーバ
  利用者のリソース保持者であり，本論文では Google，Facebook，および Twitter が該当する
- クライアント
  リソースサーバに対するアクセス権限の要求者であり，本論文では上位 500 サイトでソーシャルログインが実装されている Web アプリケーション全てが該当する

なお，OAuth2.0 にはアクセス権限委譲のフローが複数存在し，その中でも，クライアントが Web アプリケーションである場合の Authorization Code Grant を示す．

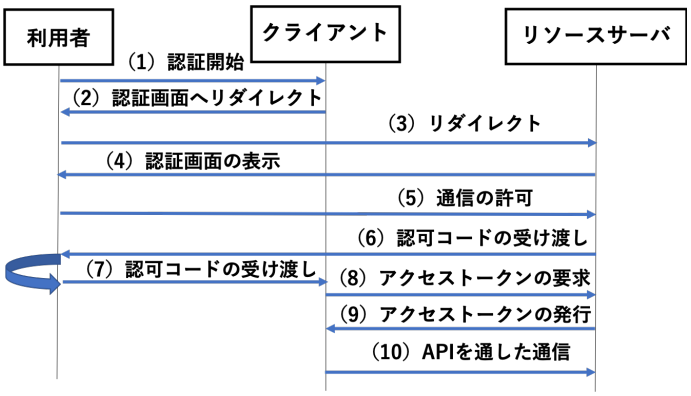

図 1 OAuth2.0 におけるアクセス権限委譲のフロー

図 1 のフローの手順を以下に示す．

(1) 利用者が認証が必要なページ（リソース）へアクセスする
(2) クライアントが利用者に，認証画面へのリダイレクト URL を送信する
(3) 利用者がリダイレクト URL へアクセスする
(4) リソースサーバが利用者のブラウザに，認証画面を提示（利用者認証の実施）
(5) 「クライアントがリソースサーバにアクセスすること」を利用者が許可する
(6) リソースサーバが利用者に認可コードを受け渡す．認可コードとは，利用者が「クライアントがリソースサーバにアクセスすること」を許可したことの証明として受け渡されるランダムな文字列である
(7) 利用者が認可コードをクライアントに受け渡す
(8) クライアントがリソースサーバにアクセストークンの発行を要求する．アクセストークンとは，リソースサーバの API にアクセスするためのトークンであり，ランダムな文字列で表現されている
(9) リソースサーバは認可コードの検証を行い，正当性が確認できれば，クライアントにアクセストークンを発行する
(10) アクセストークンを取得したクライアントは，API を通してリソースサーバ内のリソースにアクセスできる

2.2 OpenIDConnect

OpenIDConnect[6] とは，OAuth2.0 をベースとした認可フレームワークである．OAuth 同様，別のクライアントや利用者に代わり，リソースにアクセスするための方法を，クライアントに提供できる．このフレームワークは，OAuth2.0 に ID トークンのやり取りを追加したものである．

ID トークンとは，図 1 の (8) で発行されるアクセストークンとともにリソースサーバから発行され，より安全に認証を行うことを目的としている．また，ID トークンは，利用者の属性情報を含むトークンである．さらに，このトークンは，ヘッダ部，ペイロード部，および署名部に分かれており，各部分はピリオドで区切られている．各部分はヘッダ部で定められたアルゴリズムで暗号化されている．なお，ヘッダ部は base64url[7] でエンコードされ，ペイロード部には氏名，性別などの利用者の情報が JSON 形式で記述されている．そして，署名部にはヘッダ部とペイロード部に対する署名が記述されており，発行元の公開鍵を用いて復号する．

2.3 ソーシャルログインで委譲するアクセス権限

ソーシャルログインを行う際に，クライアントは利用者にサービスを利用する上で必要な権限を要求している．例えば，サービスを利用する上で利用者のメールアドレスが必要だった場合，メールアドレスを取得するための権限をクライアントが利用者に要求している．しかし，サービスを利用する上で不必要な権限を要求しているクライアントが存在する可能性が考えられる．本論文では，サービスを利用する上で必要とは限らない権限を「必要以上に要求しているアクセス権」とした．ただし，何を持って「必要」とするのかはサービス内容によって違うので，調査の際，

サービス内容より独自に判断した．

2.4 クロスサイトリクエストフォージェリ

クロスサイトリクエストフォージェリ（CSRF）[8] とは，Web アプリケーションに存在する脆弱性である．攻撃者がこの脆弱性を悪用すると，標的へ意図しないリクエストを送信する攻撃が可能になる．この攻撃例を以下に示す．

(1) 攻撃者が，意図しないリクエストを送信させる Web サイトを用意する
(2) 攻撃者が用意した Web サイトに標的を何らかの形でアクセスさせる
(3) 標的が (2) により，意図しないリクエストを Web サーバに送信する
(4) この結果，例えば標的は，インターネット上の掲示板に意図しない投稿をする，といった被害を受ける

CSRF による脅威は OAuth や OpenIDConnect においても存在する．OAuth/OpenIDConnect における CSRF[9] を悪用した攻撃例のフローを図 2 に示す．

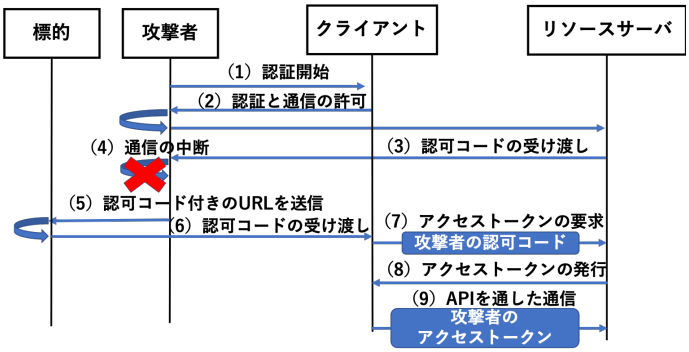

図 2 OAuth/OpenIDConnect における CSRF を悪用した攻撃例のフロー

図 2 のフローの手順を以下に示す．

(1) 攻撃者が，正規の利用者のふりをして，CSRF 脆弱性を持つ認証が必要なページへアクセスする
(2) 攻撃者，クライアント，およびリソースサーバの間で，図 1 の（2）～（5）と同様に，認証と通信の許可に関するやり取りを行う
(3) リソースサーバが攻撃者に認可コードを受け渡す
(4) 攻撃者は（3）の通信を何らかの形で中断する
(5) 中断した通信における攻撃者の認可コード付きの URL を標的に何らかの形で送信する
(6) 標的は送信された URL にアクセスし，中断された通信を標的の続きとして，続行する
(7) 標的が攻撃者の認可コードをクライアントに受け渡し，クライアントがリソースサーバにアクセストークンの発行を要求する
(8) リソースサーバは認可コードの検証を行い，正当性が確認できれば，標的に攻撃者のアクセストークンを発行する
(9) 標的の環境におけるセッションと攻撃者のアクセストークンが紐づく．つまり，標的は攻撃者の SNS アカウントでクライアントにログインしている状態である．標的が，この攻撃に気づかないままデータのやり取りを行うと，個人情報の流出などの被害に繋がることがある

OAuth/OpenIDConnect における CSRF の対策として，state パラメタ [10] を付与してフローを進める方法が推奨されている．

state パラメタは，認証を開始したセッションと認可コードを要求するセッションが，同一であるかをクライアントが判定するための文字列である．また，このパラメタは図 3 のように，リダイレクト URL のクエリに付与される．具体的には，図 2 のフローにおける (1) の後，クライアントが URL のクエリパラメタに state パラメタとして追加する．そして，クライアントが標的から，攻撃者の認可コードを受け取る際に，state パラメタが参照するセッションと現在のセッションの比較を行う．一致していた場合のみ (6) 以降の認証を進める．

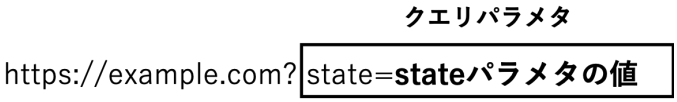

図 3 クエリパラメタと state パラメタ

3. 調査概要

3.1 調査内容

ソーシャルログインにおいて，必要以上のアクセス権の取得や実装不備により，利用者が何らかの被害を受ける可能性がある．そこで，本論文では以下の 3 点に注目し，実装状況を調査した．

- **ソーシャルログインの実装数の調査**
  どのクライアントでソーシャルログインが実装されているか調査し，その実装数を確認した．
- **クライアントが利用者に要求している権限の調査**
  クライアントが利用者に要求しているアクセス権を調査した．加えて，必要以上に要求しているアクセス権の有無を調査した．
- **CSRF 対策の実装状況の調査**
  2.4 節で示したように，ソーシャルログインに用いられている OAuth や OpenIDConnect には CSRF 脆弱性が存在する可能性がある．そこで，CSRF 脆弱性を持つ可能性のある状態で運用しているクライアント数を調査した．また，調査の際に取得した state パラメタの値と認可コードの文字列の差異を分析した．

3.2　調査対象

本論文では，2018 年 7 月時点における上位 500 サイトを調査対象とした．Top Sites in Japan - Alexa では，日本からの Web サイトへのアクセス数がランキング化されている．日本で利用されている主要な Web サイトを把握するため，このランキングを利用した．

3.3　調査方法

3.3.1　ソーシャルログインの実装数の調査

調査をするにあたり，Web サイトをクローリングするためのシステム（以降，クローリングシステムと呼ぶ）を構築し，上位 500 サイトの巡回を行なった．

クローリングシステムは，事前に用意した上位 500 サイトのログインページの URL を保存した CSV ファイルを読み込み，各 URL にアクセスする．そして，アクセス先で Google，Facebook，および Twitter 利用したソーシャルログインが存在するか判定を行う．クローリングシステムにおける判定部分のフローを図 4 に示す．

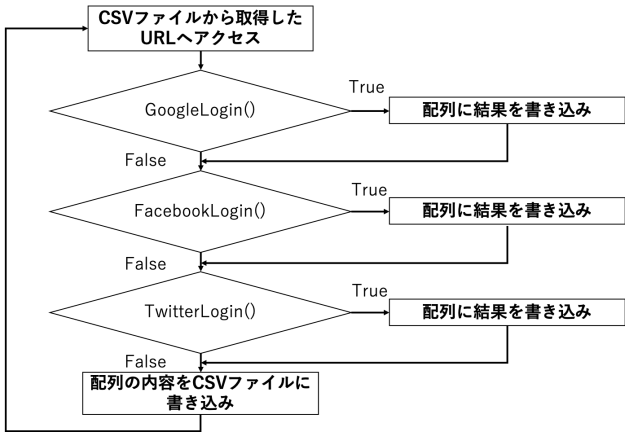

図 4　クローリングシステムにおけるソーシャルログインの有無の判定のフロー

図 4 のフローの手順を以下に示す．

(1) 用意した CSV ファイルからログインページの URL を取得しアクセスする
(2) GoogleLogin()，FacebookLogin() および Twitter-Login() を実行する．これらのメソッドは，Google，Facebook，Twitter のログインフォームにあらかじめ用意したユーザ ID とパスワードを入力しログインを実行するメソッドである
(3) GoogleLogin() のフローが完了しログインに成功した場合，そのサイトは Google のソーシャルログインを実装しているとみなす
(4) その結果を配列に書き込む
(5) （3）（4）と同様に FacebookLogin()，TwitterLogin() を実行し，結果を配列に書き込む
(6) 最終的な配列の内容を新たに用意した CSV ファイルに書き込む
(7) 始めに用意した CSV ファイルから，次のログインページの URL を取得し，（1）～（6）を上位 500 サイトに対して繰り返す

3.3.2　クライアントが利用者に要求している権限の調査

Google，Facebook，Twitter の各アカウントの設定ページから，連携されている Web アプリケーション名とクライアントに委譲された権限を確認した．

本調査では，クライアントから要求された全権限を承認したので，委譲された権限が要求している権限と同等とした．確認に用いたページの一部を Google は図 5，Facebook は図 6，Twitter は図 7 に示す．

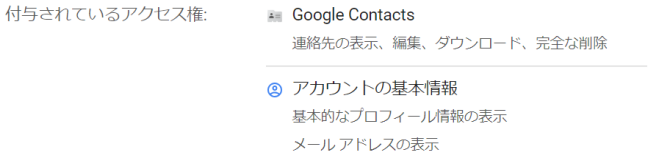

図 5　Google のアクセス権確認画面

図 5 の場合，「連絡先」，「基本プロフィール」，「メールアドレス」の権限が委譲されている．

図 6　Facebook のアクセス権確認画面

図 6 の場合，「公開プロフィール」，「友達リスト」，「出身地」，「居住地」，「写真」の権限が委譲されている．

図 7 の場合，「書き込み」，「ダイレクトメッセージ」の権限が委譲されている．

図 7　Twitter のアクセス権確認画面

3.3.3　CSRF 対策の実装状況の調査

ソーシャルログインの認可における通信を保存した．保存した通信データを確認し，認可コードを送信するリダイレクト URL に state パラメタが付与されているかを調査した．なお，state パラメタが付与されていない場合は CSRF 脆弱性を含み，付与されている場合は CSRF 脆弱性を含まないとした．また，保存した通信データから state パラメタの値と認可コードの文字列の差異を確認した．ただし，Twitter では OAuth1.0 を拡張した TwitterOAuth[11] という独自の認可フレームワークが用いられている Web アプリケーションが存在するため，調査の対象外とした．

4.　調査結果

4.1　ソーシャルログインの実装数の調査

上位 500 サイトで，Google，Facebook，Twitter のいずれかを用いたソーシャルログインが，重複も含めてどの程度実装されているかを図 8 に示す．

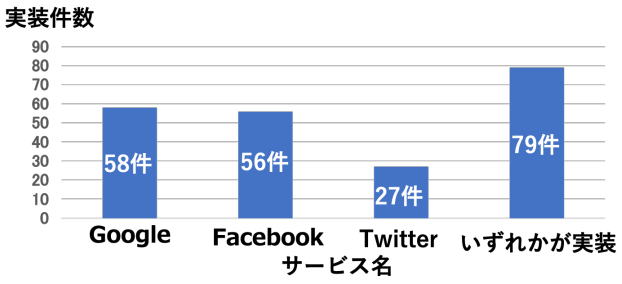

図 8　ソーシャルログインの実装数の調査

調査の結果，上位 500 サイト中 79 件でソーシャルログインの実装を確認できた．また，Yahoo!や LINE など，他のサービスを用いたソーシャルログインを確認したので，実際の割合は今回の結果より大きくなる．

4.2　クライアントが利用者に要求している権限の調査

Google，Facebook および Twitter で，クライアントが利用者に要求しているアクセス権を分類した結果を表 1，表 2，および表 3 に示す．

Google では，「メールアドレス」へのアクセス権が 58 件中 43 件，「基本プロフィール」へのアクセス権が 58 件中 40 件要求されていた．どちらの権限も，多くの場合サービスの利用上必要なので，このような結果になったと考えられる．

表 1　Google におけるアクセス権の分類と件数

付与されていたアクセス権	件数
メールアドレス	43 件
基本プロフィール	40 件
おおよその年齢	6 件
言語設定	6 件
Google で自分について検索	3 件
ドメインユーザーや組織の表示	1 件
連絡先	1 件

表 2　Facebook におけるアクセス権の分類と件数

付与されていたアクセス権	件数
公開プロフィール	50 件
メールアドレス	41 件
友達リスト	7 件
生年月日	7 件
ページへのいいね	2 件
写真	2 件
出身地	1 件
居住地	1 件

Facebook では，「公開プロフィール」へのアクセス権が 58 件中 50 件，「メールアドレス」へのアクセス権が 58 件中 41 件要求されていた．Google の場合と同様の理由で，このような結果になったと考えられる．

表 3　Twiter におけるアクセス権の分類と件数

付与されていたアクセス権	件数
読み取り	27 件
書き込み	20 件
メールアドレス	4 件
ダイレクトメッセージ	2 件

Twitter では，「読み取り」へのアクセス権が 27 件中 27 件「メールアドレス」へのアクセス権が 27 件中 4 件要求されていた．「読み取り」へのアクセス権は全ての Web アプリケーションで要求されてるのに対し，「メールアドレス」へのアクセス権の件数は Google や Facebook の場合と比較すると低い．

4.3　CSRF 対策の実装状況の調査

Google，Facebook において，state パラメタが付与されているクライアントがどの程度かを図 9 に示す．

state パラメタが付与されていないので，CSRF 脆弱性を持つ可能性があると判断できるクライアントは Google

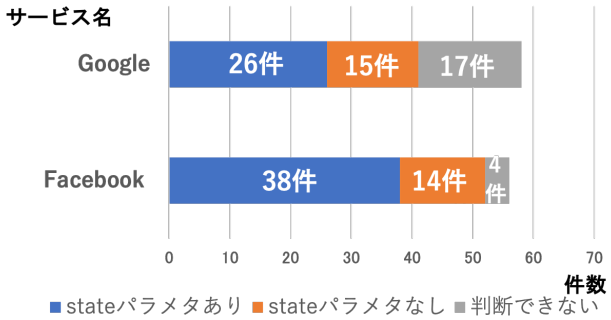

図 9　state パラメタの有無

では 58 件中 15 件，Facebook では 56 件中 14 件存在した．

また，Google では 58 件中 17 件，Facebook では 56 件中 4 件の Web アプリケーションで，認可コードを含むパケットを取得できなかったので，state パラメタが付与されているかどうか判断できなかった．

4.3.1　state パラメタの値の差異

付与されていた state パラメタの値は，以下の 3 種類に分類できることがわかった．

(1) 数字の 0 から 9 のみで構成された文字列
(2) 数字の 0 から 9，および小文字の a から f のみで構成された文字列
(3) 数字の 0 から 9，および全てのアルファベットで構成された文字列

なお，state パラメタの値に文字数の規則性は見られなかった．分類した state パラメタの値を，本論文では，(1) を 10 進表現，(2) を 16 進表現，(3) を全ての表現と呼び，Google，Facebook における state パラメタの特徴を分類した結果を表 4，表 5 に示す．

表 4　Google における state パラメタの特徴の分類

表現の種類	実装件数
10 進表現	1 件
16 進表現	10 件
全ての表現	15 件

Google では 10 進表現が 1 件，16 進表現が 10 件，全ての表現が 15 件であった．

表 5　Facebook における state パラメタの特徴の分類

表現の種類	実装件数
10 進表現	2 件
16 進表現	17 件
全ての表現	19 件

Facebook では，10 進表現が 2 件，16 進表現が 17 件，全ての表現が 19 件であった．

4.3.2　認可コードの差異

Google では 34 件，Facebook では 40 件の Web アプリケーションにおける通信で認可コードが取得可能であった．認可コードは Google では先頭の 4 文字，Facebook では先頭の 2 文字が共通していた．共通していた文字列の内容は Google では「4/jg」，Facebook では「AQ」であった．これ以降の文字列は，大文字を含むランダムな英数字で構成されており，別のアカウントでログインをした場合，および別の IP アドレスからアクセスした場合にも同様に確認できた．また，認可コードの文字数の違いを確認した．Google では，34 件中 32 件が 89 文字で構成されていたが，158 文字，184 文字で構成されているものが 1 件ずつ存在した．Facebook では，40 件中 36 件が 344 文字で構成されていたが，366 文字で構成されているものが 2 件，421 文字，および 82 文字で構成されているものが 1 件ずつ存在した．

4.3.3　アクセストークンの再利用

本論文における調査を進める中で，認可コードの代わりにアクセストークンがリダイレクト URL のクエリパラメタに付与されていた実装が存在した．その数は，Google では 58 件中 7 件，Facebook では 56 件中 5 件であった．

本来，アクセストークンの漏洩を防ぐために有効期間の短い認可コードがリダイレクト URL のクエリパラメタに付与されている．しかし，アクセストークンをリダイレクト URL のクエリパラメタに付与する実装により，アクセストークンが簡単に漏洩してしまうリスクが高まり，情報漏洩やなりすましなどに悪用される危険性があると言える．

5.　考察

5.1　Web アプリケーションが利用者に要求している権限

5.1.1　Google 利用におけるアクセス権限

Google では，「連絡先」への権限を要求するクライアントが 58 件中 1 件存在した．

この Web アプリケーションは利用者にオンラインのストレージを提供するサービスであった．この Web アプリケーションを運営している企業によると，利用者の友人にサービスを紹介するメールや，ファイルを共有するリンクを含むメールを送信するために，「連絡先」へのアクセス権限を要求するとしている．

しかし，利用者は，連絡先の連携により生じるリスクの大きさを理解していない可能性がある．よって，連絡先を連携する場合は，他の権限を与える時よりも目立つ形で利用者に説明することが望ましい．

さらに，要求されていたアクセス権限の中で，「Google で自分について検索」や「ドメインユーザー，組織の表示」といった，利用者が内容を理解しにくいものが存在した．これらのアクセス権限が付与されることで，クライアントがどのようなリソースへアクセス可能かについて，Web ア

プリケーション作成者からの記述はなかった．これらの権限がサービスを利用する上で，どのように必要なのか判断できないので，利用者の理解を促す説明があることが望ましい．

5.1.2 Facebook 利用におけるアクセス権限

Facebook では，「友達リスト」への権限を要求するクライアントが 56 件中 7 件存在した．友達リストを閲覧する権限が与えられている Web アプリケーションでは，多くの場合，利用者の友人へサービスの紹介を目的としている．

しかし，Facebook は実名登録を規約としており，出身地や出身校なども登録できるので，Google，Facebook，および Twitter の中で個人を特定し得る情報を最も多く持つ．つまり，クライアントに友達リストを閲覧する権限を与え，悪用された場合のリスクが大きいと考えられる．Facebook のみ，図 6 のように，利用者が連携するアクセス権限を設定できる．よって，利用者が不必要だと判断したアクセス権限を付与しないよう設定することで，権限付与を必要最小限に抑える工夫の余地がある．

5.1.3 Twitter 利用におけるアクセス権限

Twitter では，「ダイレクトメッセージ」，「書き込み」への権限を要求するクライアントが 27 件中それぞれ 20 件，4 件存在していた．

Twitter 社は，アカウントの乗っ取りが疑われる現象の例として『アカウントから心当たりのないツイート』，『アカウントから心当たりのないダイレクトメッセージが送信』などを挙げている [12]．その原因として，「書き込み」と「ダイレクトメッセージ」の権限を与えられたクライアント側が原因となる可能性がある．さらに，「ダイレクトメッセージ」の権限を要求していた 2 つのクライアントには，この権限の利用目的に関する記述はなかった．

クライアントは，求める権限を最小に留め，利用者に選択肢を提供し，十分な説明をすることが望ましい．

5.1.4 必要以上に要求しているアクセス権

今回の調査で，Google と Facebook において，利用者の友人などに招待メールを送信することを目的としてアクセス権を要求するクライアントが 8 件存在した．

しかし，利用者が友人にサービスを紹介するために，クライアントにその友人の連絡先を渡す必要があるとは限らない．例えば，招待 URL を発行し，利用者が友人に何らかの手段でその URL を送信する方法が考えられる．これにより，サービスの紹介機能は実現できる．したがって，連絡先へのアクセス権は必要とは限らないので，必要以上に要求しているアクセス権とみなした．

さらに，Google において，利用者が内容を理解できないアクセス権を要求しておるクライアントが 4 件存在した．また，Twitter において，利用目的が不明であるアクセス権を要求しているクライアントが 2 件存在した．これらのアクセス権は必要でない可能性が高いので，必要以上に要求しているアクセス権とみなした．

5.2 CSRF 対策の実装状況の調査について

4.3 節の結果より，CSRF 脆弱性を考察する．

5.2.1 state パラメタの付与状況について

調査の結果，state パラメタが付与されていない状態で運用されている可能性のあるクライアントが Google では 58 件中 15 件，Facebook では 56 件中 14 件存在していた．state パラメタが付与されていないクライアントは，CSRF 脆弱性が存在している可能性があり，危険である．

多くのクライアントで state パラメタが付与されていない理由は 2 つ考えられる．1 つ目は，RFC6749[2] で付与を必須としていないことである．state パラメタの付与は推奨されているだけなので，その重要性を Web アプリケーション作成者が理解していない可能性がある．2 つ目は，実装自体の難易度が高いことが理由として，考えられる．5.2.2 節でも示すが，state パラメタを付与することの重要性を理解していても実装が困難な可能性がある．

よって，Web アプリケーション作成者がこの脆弱性を修正することを期待するだけでなく，state パラメタの付与を必須とし，その実装方法をより簡潔に記載することが期待される．

5.2.2 state パラメタの値の差異について

調査の結果，付与されている state パラメタの値の文字列が，10 進表現，16 進表現および全ての表現が存在した．state パラメタ値の文字数に規則性が見られなかったことより，state パラメタの実装が全て Web アプリケーション作成者に依存していることが推察される．これは，攻撃者によるパラメタ値の推測などによる攻撃も想定され，実装によっては，危険な状況にあるとも言える．

state パラメタの実装が危険な状況だと，少なからずのソーシャルログインを実装する開発者らが，OAuth のセキュリティガイドライン [4] を参照して，安全な実装をしていない可能性も想定できる．

さらに，本論文では CSRF 脆弱性に注目し調査したが，文献 [13][14][15] に示されている通り，OAuth や OpenID-Connect には多くの脆弱性が指摘されている．これらへの対処も実装者に委ねられていると言え，さらなる調査が必要であろう．

5.2.3 アクセストークンの再利用について

認可コードの代わりにアクセストークンが再利用され，リダイレクト URL のクエリパラメタに付与されていた実装が Google では 58 件中 7 件，Facebook では 56 件中 5 件見つかった．

クエリパラメタに付与した値は，簡単に入手されてしまう可能性がある．例えば，Yuchen らの調査 [15] に示されているように，利用者のタイムラインへの投稿や機密情報へのアクセスなどのなりすましの被害に遭う可能性がある

ことが指摘されている．

したがって，アクセストークンのような識別情報をクエリパラメタに付与することは避けるべきである．

6. 研究倫理

本論文では，調査対象のサービスに対して悪影響を及ぼさないように注意して実験を行った．また，論文を執筆する際は上位 500 サイト中の特定のサービス名を特定できないように配慮した．

7. まとめ

本論文では，クローリングシステムを用いて上位 500 サイトを巡回し，ソーシャルログインの実装状況を調査した．

その結果以下の 6 点が明らかになった．

(1) 上位 500 サイト中 79 サイトにソーシャルログインが実装されていた

(2) 必要以上にアクセス権を要求している判断できるクライアントが Google では 5 件，Facebook では 7 件，Twitter では 2 件存在した

(3) state パラメータが実装されていないので CSRF 脆弱性が存在している可能性のあるクライアントが Google では 15 件，Facebook では 14 件存在した

(4) state パラメータが実装されている場合，その表現がクライアントごとに異なっていた

(5) Google と Facebook 双方の認可コード長に違いがあり，先頭の数文字は共通していることがわかった

(6) クエリパラメータにアクセストークンを再利用して送信する実装をしているクライアントが Google で 7 件，Facebook で 6 件存在していた

以上より，問題のある Web アプリケーションが国内に存在することが確認された．

参考文献

[1] RFC5849 The OAuth 1.0 Protocol. https://tools.ietf.org/html/rfc5849.

[2] RFC6749 The OAuth 2.0 Authorization Framework. https://tools.ietf.org/html/rfc6749.

[3] RFC6750 The OAuth 2.0 Authorization Framework: Bearer Token Usage. https://tools.ietf.org/html/rfc6750.

[4] RFC6819 OAuth 2.0 Threat Model and Security Considerations. https://tools.ietf.org/html/rfc6819.

[5] Top Sites in Japan - Alexa. https://www.alexa.com/topsites/countries/JP.

[6] OpenID Connect Core 1.0 incorporating errata set 1. http://openid-foundation-japan.github.io/openid-connect-core-1$_0$.*ja.html*.

[7] RFC4648 The Base16, Base32, and Base64 Data Encodings . https://tools.ietf.org/html/rfc4648.

[8] CWE-352: Cross-Site Request Forgery (CSRF). https://cwe.mitre.org/data/definitions/352.html.

[9] Threat: CSRF Attack against redirect-uri. https://tools.ietf.org/html/rfc6819section-4.4.1.8.

[10] The oauth 2.0 authorization framework 4.1.1. authorization request.

[11] TwitterOAuth. https://twitteroauth.com/.

[12] Twitter アカウントが乗っ取られた場合のヘルプ. https://help.twitter.com/ja/safety-and-security/twitter-account-compromised.

[13] Vladislav Mladenov, Christian Mainka, Julian Krautwald, Florian Feldmann, and Jörg Schwenk. On the security of modern single sign-on protocols: Openid connect 1.0. *CoRR*, Vol. abs/1508.04324, , 2015.

[14] Manuel Urueña, Alfonso Muñoz, and David Larrabeiti. Analysis of privacy vulnerabilities in single sign-on mechanisms for multimedia websites. *Multimedia Tools and Applications*, Vol. 68, No. 1, pp. 159–176, Jan 2014.

[15] Yuchen Zhou and David Evans. Ssoscan: Automated testing of web applications for single sign-on vulnerabilities. pp. 495–510, 2014.