Look at the source: refine standard to harden OAuth security

Tommaso Innocenti
innocenti.t@northeastern.edu
Tommaso Innocenti
PhD candidate
@Northeastern University (Boston)
innotommy.com
What the heck is OAuth 2.0?

Introduction

Bob

Service Provider

Identity Provider

Signs in with IMDb

Sign in with Amazon

Sign in with Google

Sign in with Apple

Show more options

Create a New Account

Visits Website

Login Request

User Authentication and Consent

Login Response

"Hi, Bob!"
"redirect_uri" attacks timeline

2012
RFC 6749 The OAuth 2.0 Authorization Framework

2013
RFC 6819 - OAuth 2.0 Threat Model and Security Considerations

2014
Covert Redirect

2017
OAuth 2.0 Security Best Current Practice

2019
Make Redirection Evil Again

2022
OAuth 2.0 Redirect URI Validation Falls Short, Literally

2023
OAuch: Exploring Security Compliance in the OAuth 2.0 Ecosystem

Code grant flow

1) Client Application Access
2) Redirection to IdP Login
3) Authorization Request
   [response_type=code, client_id, client_secret, code, state=redirect_uri]
4) User Authentication
   Parameter Validation
5) Redirection to Client Callback
6) Authorization Response
   [code, state]
7) Access Token Request
   [grant_type=authorization_code, client_id, client_secret, code, state=redirect_uri]
8) Access Token Response
   [access_token]
9) Protected Resource Requests
   [access_token]
10) Protected Resource Response
    [Data]
• Current research trend analysis
  • Covert redirect
  • Make Redirection Evil Again

• Look at the "source" concept
  • Expert validation
  • impact

External validation

Our concept
redirect_uri validation in RFC:

- **RFC 6749 Section 3.1.2.3**
  The authorization server **MUST** compare the two URIs using simple string comparison as defined in RFC 3986 Section 6.2.1.

- **RFC 3986 Section 6.2.1**
  Testing strings for equality is normally based on pair comparison of the characters that make up the strings, starting from the first and proceeding until both strings are exhausted, and all characters are found to be equal, until a pair of characters compares unequal, or until one of the strings is exhausted before the other.
What is Path Confusion?

PathConfusion:

```
../../../FAKEPATH
%/252e%252e%252FFAKEPATH
```

Bob visits the IMDb website. In the login process, the user’s browser sends a login request to the IMDb server. However, a malicious website somewhere on the internet sends the same login request to the IMDb server using a URL that looks like it is coming from IMDb, but it is not. This is an example of path confusion, where a malicious website impersonates a legitimate website to gain user trust and access to sensitive information.

The identity provider is also involved in the process of verifying the user’s identity and granting access. The service provider, in this case, IMDb, processes the login request and authenticates the user. However, if the malicious website is able to intercept or manipulate the communication, it can steal user credentials.

The diagram illustrates the flow of events, highlighting the potential for path confusion to occur.
Verify our hypothesis

• Find empirical validation

DISTINCT (CCS’22)[1]  Manual login + In browser communications
Cerberus (CCS’22)[2]  Static analysis of libraries
SAAT (S&P’22)[3]  Limited to only Facebook + session management
MoScan (ISSTA’21)[4]  State machine analysis
OAuthScope (WPES’21)[5]  Limit to 4 providers + user’s data access

Modular, scalable and capable

Methodology

Setup
- Sites & OAuth 2.0 Triggers
- IdP Detection
- Login page detection
- Tranco sites list
- IdP Credentials

Data Collection
- OAuth 2.0 Player
- IMDb.com/LOGIN
- Facebook.com/LOGIN
- Sign in
- Username
- ****
- Path
- Confusion
- Payloads
- IMDb
- Facebook
- mitmproxy

Data Analysis
- Network Dump
- Login Results
- Proxy Logs
- Screen Captures
- OAuth 2.0 Flow Analysis
- Analysis Results

https://github.com/innotommy/OAuthpaper-code
6/16 IdPs vulnerable to Path Confusion
(Facebook, Microsoft, GitHub, Atlassian, NAVER, and VK)
Are we doomed?

1) Client Application Access
2) Redirection to IdP Login
3) Authorization Request
4) User Authentication
5) Redirection to Client Callback
6) Resource Owner
7) Access Token Request
8) Access Token Response
9) Protected Resource Requests
10) Protected Resource Response

Client

Authorization Process

State Validation

Redeem Process

Data Access

Identity Provider

Parameter Validation

Access Token Validation

Authorization Response

(code, state)

User Agent
(Web Browser)

[response_type=code, client_id, state, redirect_uri]

[grant_type=authorization_code, client_id, client_secret, code, redirect_uri]

[access_token]

[access_token]

[Data]
**redirect_uri** validation in redeem step

- RFC 6749 Section 4.1.3
  The Client makes a request to the token endpoint by sending the following parameters[...] *redirect_uri* REQUIRED, if the "redirect_uri" parameter was included in the authorization request as described in Section 4.1.1, and their values MUST be identical.

Will IdPs use the same vulnerable validation procedure?

2/16 IdPs are vulnerable (Naver, GitHub)

Full Victim’s account takeover is possible!!!
• Path Confusion

Attack checklist:
1) Vulnerable *redirect_uri* parsing in Authorization step → 6/16 IdPs ✓
2) Vulnerable Client → openbugbounty.com ✓
3) Vulnerable *redirect_uri* check in redeem step → 2/16 IdPs ✓

Attack URL:


Full Victim’s account takeover is possible!!!
All IdPs involved in the study which has been found vulnerable has been contacted.

• Microsoft acknowledge our report and fixed their validation procedure.
• GitHub is tracking internally the problem and is actively working on a fix
• We are actively working with Naver to help fixing the issue

Reported our findings to the OAuth working group, which included our recommendation in the BCP.
OpenID foundation modified the conformance test suite to include our attack
Current “best practice” is not good enough

Recommendations:

1) `redirect_uri` validation should use strict string equality check

2) IdPs server should never sanitize `redirect_uri` to avoid introducing any discrepancy, instead should validate them
Summary and discussion

Our concept takes time, effort and doesn’t bring money

- February 2023 first contact with OAuth WG → August 2023 OSW → November 2023 BCP Acknowledged

OpenID conformance test suite updated

Follow the current trend or go back at the source?
Questions?