

REPORT DOCUMENTATION PAGE					Form Approved OMB No. 0704-0188	
<p>The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.</p> <p>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</p>						
1. REPORT DATE (DD-MM-YYYY)		2. REPORT TYPE			3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE				5a. CONTRACT NUMBER		
				5b. GRANT NUMBER		
				5c. PROGRAM ELEMENT NUMBER		
6. AUTHOR(S)				5d. PROJECT NUMBER		
				5e. TASK NUMBER		
				5f. WORK UNIT NUMBER		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)					8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)					10. SPONSOR/MONITOR'S ACRONYM(S)	
					11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT						
13. SUPPLEMENTARY NOTES						
14. ABSTRACT						
15. SUBJECT TERMS						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON	
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (Include area code)	

Overview of OAuth and OpenID Connect (OIDC)

**Beth Abramowitz, Kelley Burgin, Neil McNab,
Mike Peck, Mark Russell, Roger Westman**

April 2021

MITRE | SOLVING PROBLEMS
FOR A SAFER WORLD™

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE Public Release 21-1441.

Background

- **OAuth 2.0 – Authorization Framework**

- IETF RFC 6749 – October 2012
- Allows user to delegate access to resources
- OAuth 2.1 update in progress: current best practices, simplification

- **OpenID Connect - Federated Authentication**

- Published by OpenID Foundation
- Extends OAuth 2.0

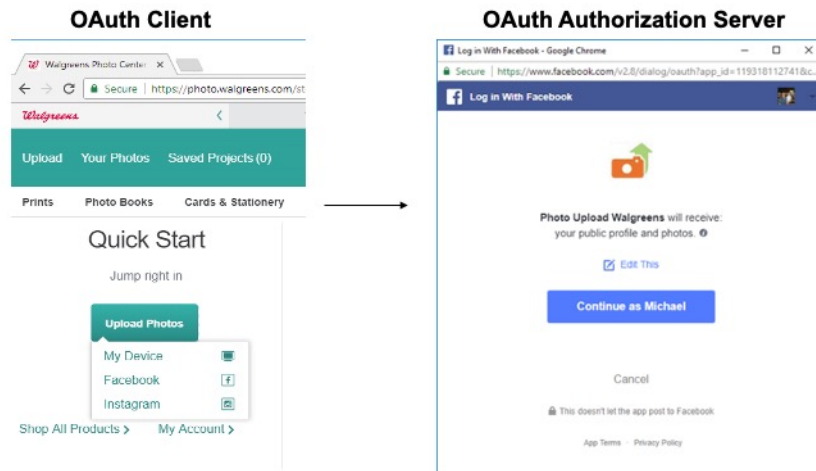
- **Wide adoption: Facebook, Google, Microsoft, Twitter, etc.**

- OAuth: API access to Gmail, Google Drive, Office 365 Email/OneDrive, etc.
- OpenID Connect: Use as Identity Provider for authentication

Background

- **Extensions/profiles to OAuth 2.0 published over the years**
 - Add capabilities, address security issues, profile for environments (e.g., banking)
 - MITRE-developed profiles for USG / enterprise environments
 - OAuth Identity Bridging & OpenID Connect profiles – February 2020
 - Socializing with standards bodies/vendors
 - Update in progress
 - OAuth Identity Chaining Profiles
 - Drafts under review

Example OAuth Use Case: Web Application



Walgreens gets access to user's Facebook photos, but nothing else

OAuth Mission Specific Example

- **Delegate authorization to an entity to act on a user's behalf**

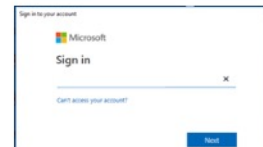
- **Example: Allow Front-End Web Server to access Back-End Database on user's behalf**
 - Front-End Web Server is issued an access token after both the user and the web server authenticate to an Authorization Server
 - Front-End Web Server uses access token to access Back-End Database
 - Back-End Database provides access only to data that both the front-end web server and the user are authorized to receive

Example OAuth Use Case: Native Application

OAuth Client



OAuth Authorization Server




Outlook app gets access to user's email, calendar, etc.

Instead of storing a username/password, stores OAuth tokens


Example Uses of OpenID Connect


Sign in with your Grubhub account

☐ Keep me signed in [Reset password](#)

 Sign in

or

 Continue with Facebook

 Connect your Google account

Welcome to Zillow

Sign in

New account

Email

Enter email


Password


Enter password


Sign in

Forgot your password?

Or connect with:


 Continue with Apple


 Continue with Facebook


 Continue with Google


medium.com


Welcome back.

 Sign in with Google

 Sign in with Facebook

 Sign in with Apple

 Sign in with Twitter

 Sign in with email

No account? [Create one](#)

Click "Sign in" to agree to Medium's Terms of Service and acknowledge that Medium's Privacy Policy applies to you.

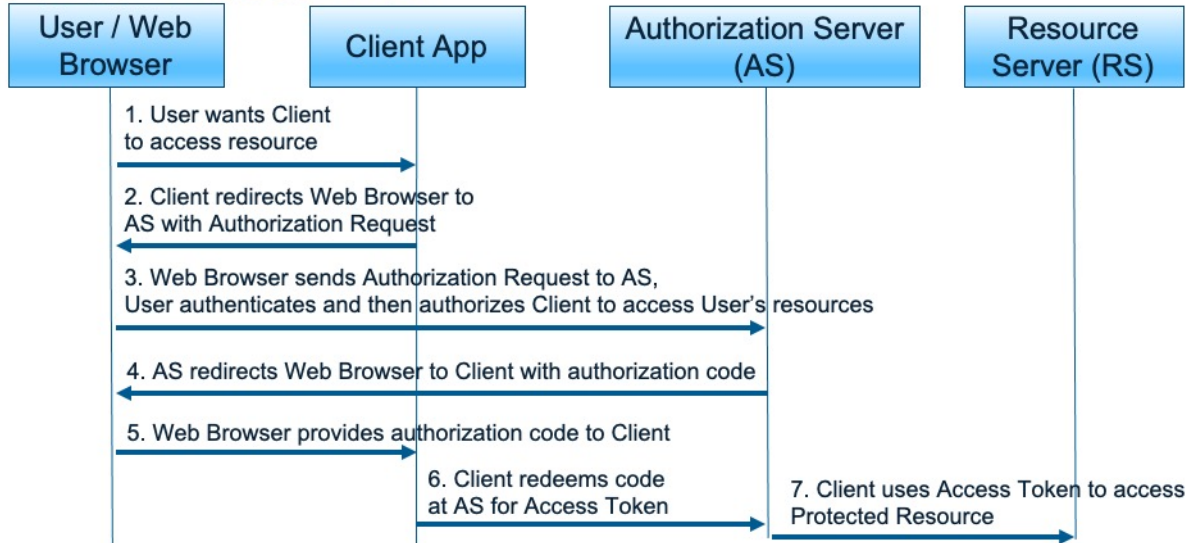
User authentication delegated to an Identity Provider

MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

7

OAuth Protocol Flow

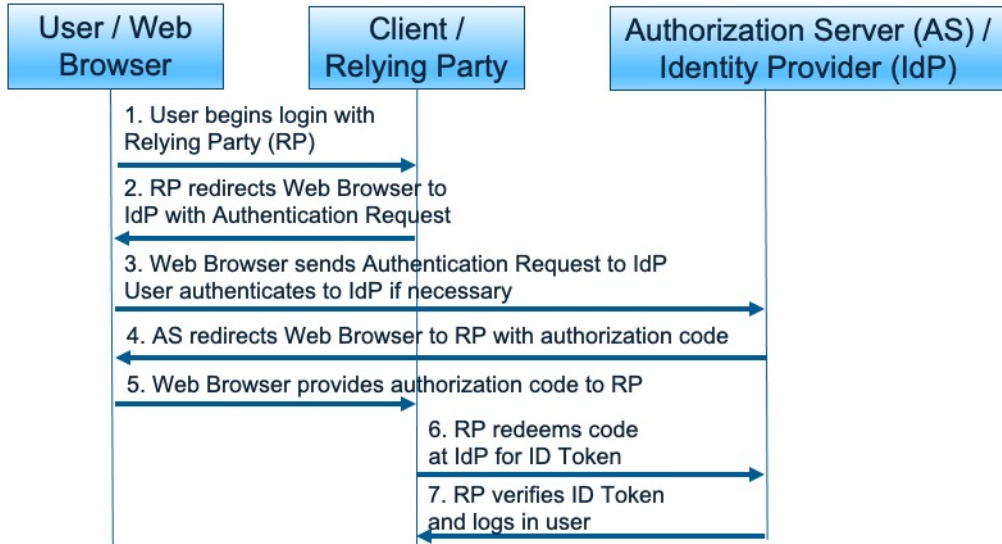


MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

8

OpenID Connect Protocol Flow



Advantages of using OAuth and OpenID Connect (1 of 2)

- **Authentication abstracted away from app**
 - Don't need to modify app to support new authentication methods
 - TLS client certs, FIDO, RSA SecurID or other OTPs, etc.
 - Avoid need to keep trusted CA list up to date at each Relying Party
 - Can dynamically adapt required authentication methods, add step-up auth
 - Can incorporate device identity and posture into access control decisions
- **Can support access control decisions based on identity of both the user and the application**
 - e.g., Compute intersection of the user's and the application's privileges



MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

10

Advantages of using OAuth and OpenID Connect (2 of 2)

- **Application does not need to know or store user's account credentials (username/password, PKI private key, etc.)**
- **Application can't arbitrarily impersonate users**
 - User involved in authorizing access to resources
 - Increased auditability
- **Scope of authorization can be limited to subset of user's access**
 - Application only gets access to what it needs (if properly implemented)
- **Can revoke access of individual clients without revoking everyone's access or forcing the user to reset credentials**

OAuth Grant Types

- **Method by which the OAuth Client obtains an Access Token**

- Authorization Code Flow becoming most commonly used; Recommended best practice.
- Implicit Flow and Resource Owner Password Credentials no longer considered appropriate: OAuth 2.1 will officially deprecate. Implicit flow may be still used by some applications.



Other flows exist too: e.g., Device Flow

User authenticates & authorizes access on a separate device than the one getting the token

Credit: Aaron Parecki, Okta - <https://developer.okta.com/blog/2019/02/19/add-oauth-device-flow-to-any-server>

OAuth Client Authentication to Authorization Server

- **OAuth supports authentication of the client separate from user authentication**
- **Two types of OAuth clients**
 - Confidential – can hold a secret – e.g., web app clients
 - Public – can't hold a secret – e.g., native app clients
- **Confidential clients authenticate to the Authorization Server when obtaining tokens**
 - Typically, a “client secret” (e.g., password) is used
 - IETF RFC 8705 describes use of TLS Client Certificate Authentication
- **Public clients do not authenticate**
 - Means there is less confidence in identity of public clients

OAuth Access Token Format and Protection

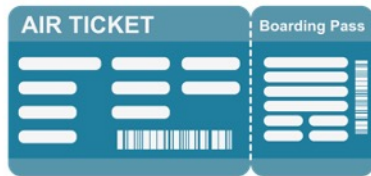
- **RFC 6749 does not define a format for access tokens**
- **Authorization Server and Resource Server must agree on format to interoperate – usually AS and RS are operated by the same entity**
- **JSON Web Token (JWT) commonly but not always used**
 - Defines a format
 - Defines cryptographic protection
 - Still need to define organization-specific claims
- **Opaque tokens can be used too**
 - Resource Server reaches back to Authorization Server to verify and interpret token

Types of Access Tokens



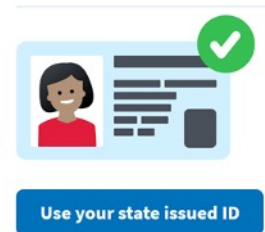
Bearer Token

Anyone can present the token
Subject to theft, replay, duplication
Recommend: Avoid if practical to do so



Sender-Constrained Token

Only an authorized entity
can use the token - must provide
proof
(e.g., PKI client authentication)
Recommend: Use when practical



MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

15

<https://www.flickr.com/photos/yum9me/495935725>

<https://pixabay.com/illustrations/ticket-trip-business-immigration-5957841/>

<https://www.dmv.virginia.gov/drivers/#newlook.asp>

Profiling OAuth and OpenID Connect

- **Problem:** IETF RFC 6749 (OAuth 2.0) is a framework – not sufficient by itself
 - Provides many options to choose from
 - Deliberately does not define some areas necessary for interoperable implementations
 - Difficult to track the many IETF RFCs addressing security issues and adding extensions
 - Need implementation guidance tailored to enterprise environments
- **MITRE's Enterprise Mission Tailored OAuth 2.0 and OpenID Connect Profiles**
 - OAuth profile describes “identity bridging” with two use cases
 - OpenID Connect describes federated authentication
 - Requirements for Clients, Protected Resources, and Authorization Servers for implementation of RFC 6749 and accompanying specifications

Profiling OAuth and OpenID Connect: Related Work

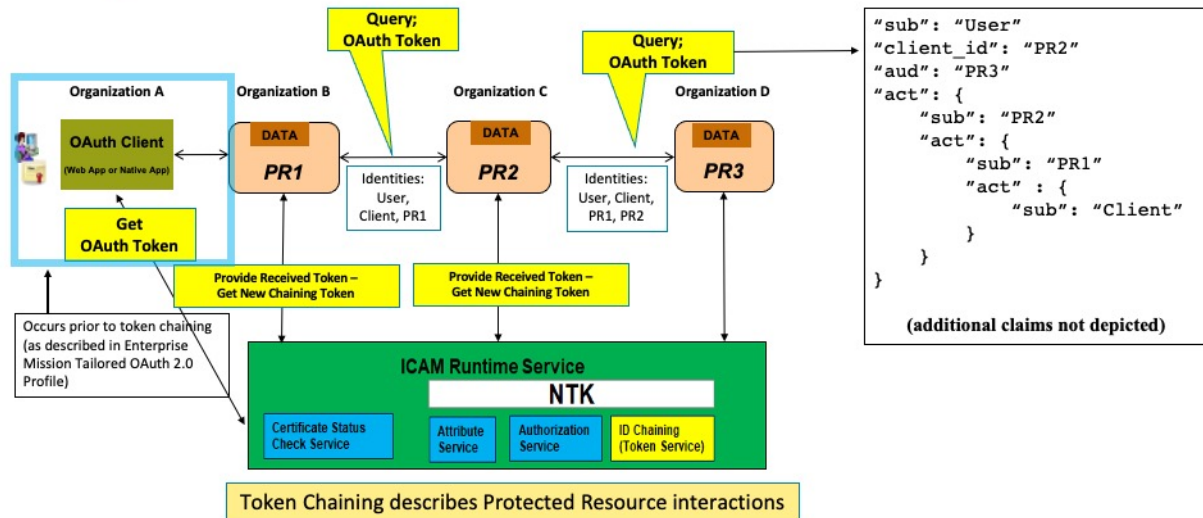
- **IETF RFC 6819: OAuth 2.0 Threat Model and Security Considerations (January 2013)**
- **IETF RFC 8252: OAuth 2.0 for Native Apps Best Current Practice**
- **IETF Draft: OAuth 2.0 Security Best Current Practice (in progress)**
- **OpenID Foundation FAPI (Financial-grade API) versions 1 and 2**
- **IETF Draft: OAuth 2.1 (in progress)**
 - **Goal: Incorporate OAuth security best practices and extensions into one RFC**

Currently Updating MITRE OAuth Profile (Identity Bridging)

- **Incorporate editorial corrections, address received comments, update references, etc.**

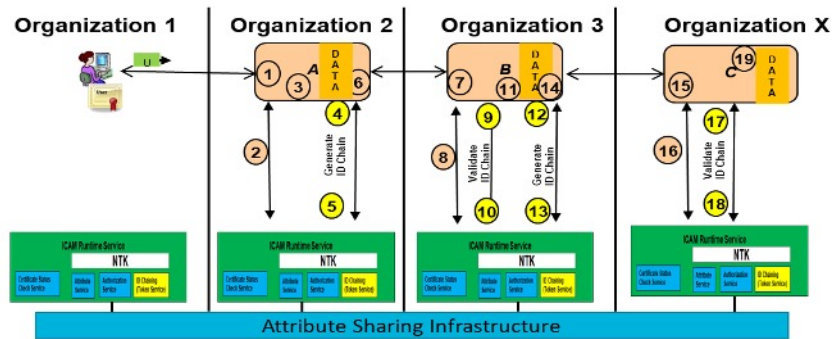
- **Simplify by aligning with OAuth 2.1 instead of OAuth 2.0**
 - OAuth 2.1 has similar goals as our OAuth profile
 - Compared our profile's requirements with OAuth 2.1 draft
 - Removed content in our profiles already addressed by OAuth 2.1
 - Submitting suggestions to IETF for OAuth 2.1

Upcoming MITRE Token Chaining Profiles: Single ICAM Ecosystem



Upcoming MITRE Token Chaining Profiles: Multiple ICAM Ecosystem

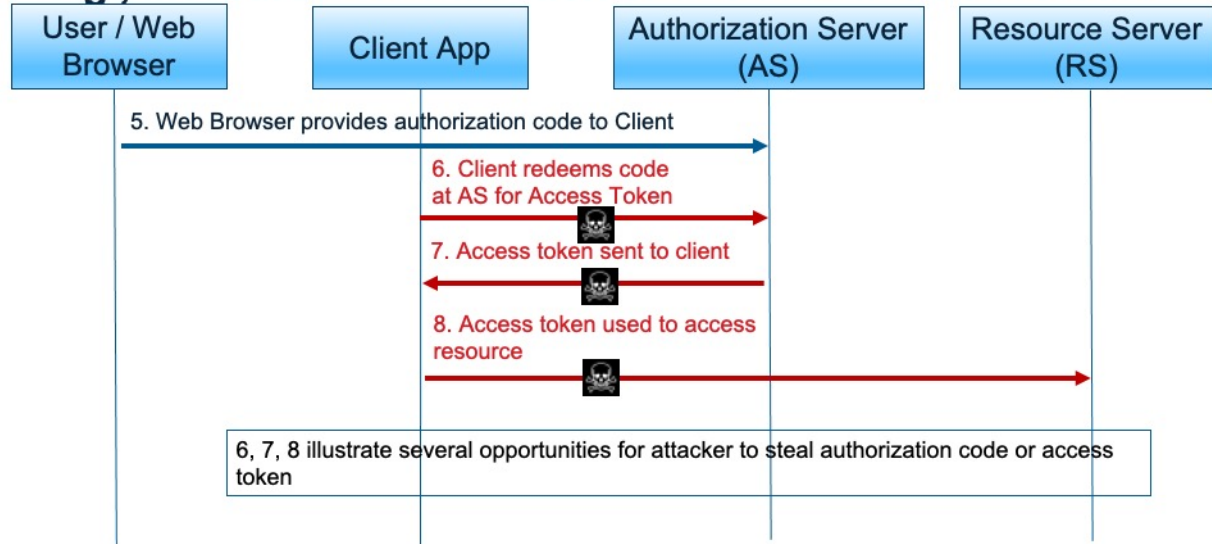
ID Chaining Tailored Profile:
Discrete ICAM Eco-Systems
(aka Multi-Organization)



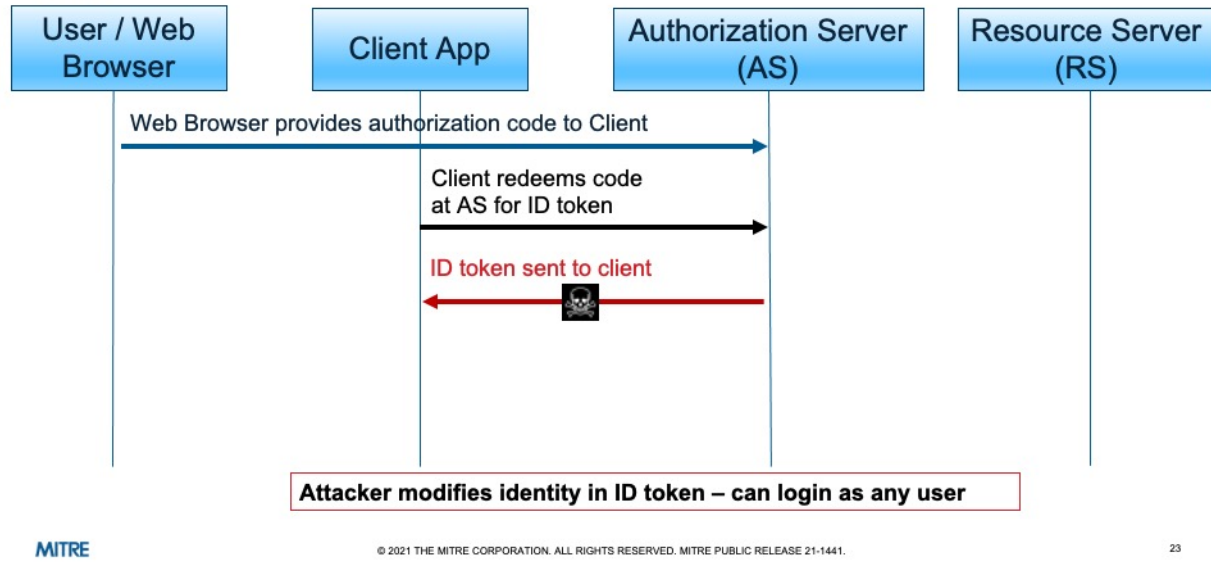
Examples of OAuth / OpenID Connect Attacks and Mitigations

- **Protocol issues**
- **Implementation Issues**

Insecure Network Communication on Client e.g., Insecure Cert Validation or use of Plaintext Protocols



Insecure Network Communication on Client (OpenID Connect variant)

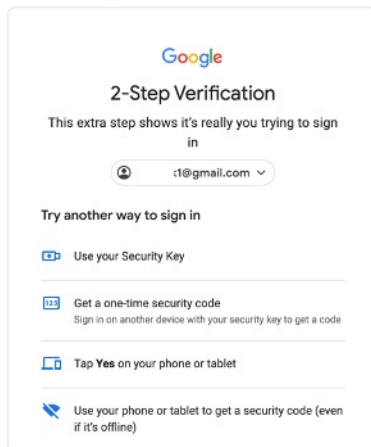


Insecure Network Communication Mitigations

- **Use HTTPS everywhere and test implementations**
- **IETF RFC 7636 Proof Code for Key Exchange (PKCE)**
 - Ensure that only the client that requests an authorization code can redeem it
- **IETF RFC 8705 Client Certificate Authentication to Authorization Server**
 - Ensure that only the valid client can redeem authorization code
- **IETF RFC 8705 Certificate-Bound Access Tokens**
 - Cryptographically bind issued access token to PKI certificate associated with private key held by client
- **OpenID Connect: Cryptographically signed ID tokens (JWT)**

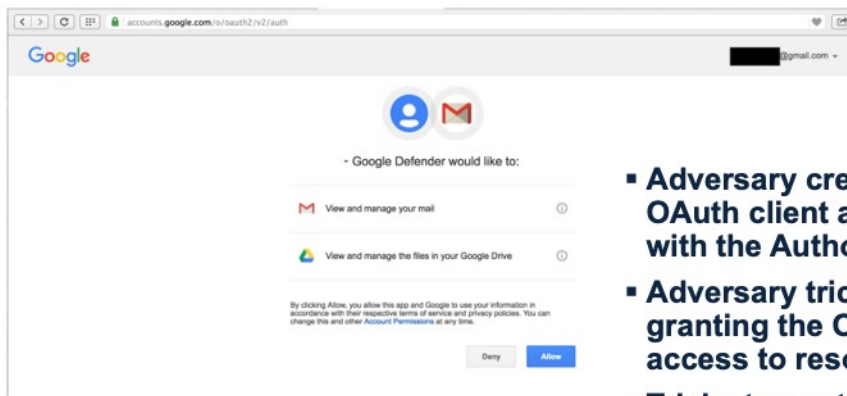
OAuth Consent Phishing

- As major web sites adopt strong multi-factor authentication... adversaries turn to OAuth to gain access...



The screenshot shows the Google 2-Step Verification interface. At the top is the Google logo, followed by the heading "2-Step Verification". Below this, a message states: "This extra step shows it's really you trying to sign in". A dropdown menu shows the email address "t1@gmail.com". Under the heading "Try another way to sign in", there are four options, each with a corresponding icon and a horizontal line below it: 1. "Use your Security Key" with a blue key icon. 2. "Get a one-time security code" with a blue code icon; a sub-note says "Sign in on another device with your security key to get a code". 3. "Tap Yes on your phone or tablet" with a blue phone icon. 4. "Use your phone or tablet to get a security code (even if it's offline)" with a blue phone icon.

OAuth Consent Phishing



- Adversary creates a “fake” OAuth client and registers it with the Authorization Server
- Adversary tricks user into granting the OAuth client access to resources
- Tricky to spot – request comes from the real provider

Credit: Trend Micro
https://www.trendmicro.com/en_us/research/17/d/pawn-storm-abuses-open-authentication-advanced-social-engineering-attacks.html

MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

26

References/examples:

https://www.trendmicro.com/en_us/research/17/d/pawn-storm-abuses-open-authentication-advanced-social-engineering-attacks.html

<https://www.proofpoint.com/us/blog/threat-insight/ta2552-uses-oauth-access-token-phishing-exploit-read-only-risks>

<https://www.microsoft.com/security/blog/2020/07/08/protecting-remote-workforce-application-attacks-consent-phishing/>

<https://security.googleblog.com/2017/05/protecting-you-against-phishing.html>

https://www.reddit.com/r/google/comments/692cr4/new_google_docs_phishing_scam_almost_undetected/

<https://www.menlosecurity.com/blog/from-your-account-is-deactivated-to-oauth-the-evolution-of-phishing>

OAuth Consent Phishing Mitigations

Ensure OAuth client registration/use is a controlled process

For example, Google and Microsoft provide admin controls to manage use of 3rd party OAuth clients

Google for example now requires 3rd party certification of OAuth clients in some cases:

OAuth API verification FAQs

Last modified on: March 12, 2021

If your app uses Google APIs to access Google users' data, you might have to complete a verification process before you publish your app.

<https://support.google.com/cloud/answer/9110914?hl=en>

MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

27

References/examples:

https://www.trendmicro.com/en_us/research/17/d/pawn-storm-abuses-open-authentication-advanced-social-engineering-attacks.html

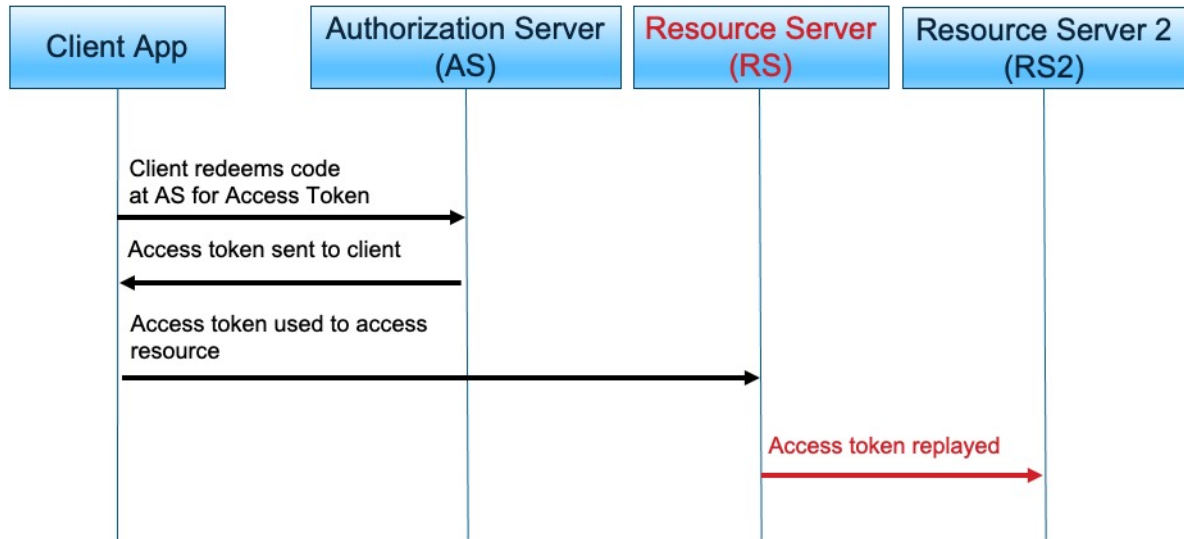
<https://www.proofpoint.com/us/blog/threat-insight/ta2552-uses-oauth-access-token-phishing-exploit-read-only-risks>

<https://www.microsoft.com/security/blog/2020/07/08/protecting-remote-workforce-application-attacks-consent-phishing/>

<https://security.googleblog.com/2017/05/protecting-you-against-phishing.html>

<https://support.google.com/cloud/answer/9110914?hl=en>

Replay of Access Tokens



Replay of Access Tokens

- **Mitigations**

- Sender-constrain access tokens
 - Bind token to client's certificate, only allow use of mutually authenticated TLS
- Audience-constrain access token
 - Put resource server's identity in access token
 - Resource server verifies that it is intended recipient

Open Redirects

Authorization Servers generally require each client register its Redirect URIs

Ensure that Authorization Code is sent to the real client, not an attacker

Potential Weakness: Authorization Server may allow wildcard values to be specified for Redirect URI

e.g., *.facebook.com

Many web sites contain open redirects, making wildcards dangerous

Web page that takes a destination page as a parameter, helps user get there

Under some conditions, attacker may be able to modify Redirect URI in OAuth request to another value allowed by the wildcard

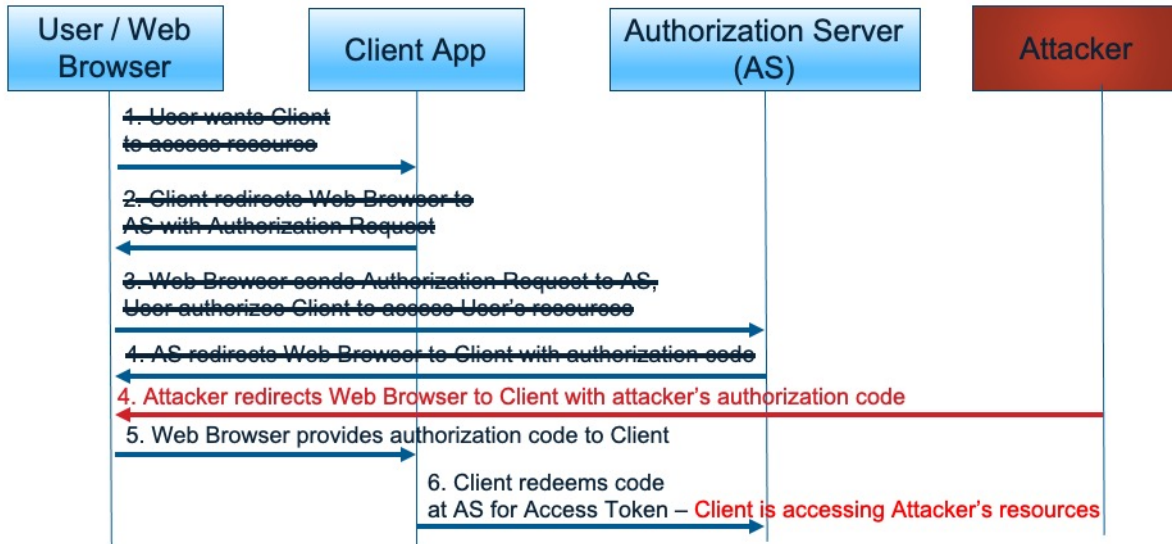
In implicit flow, results in access token being sent to attacker

In authorization code flow, results in authorization code being sent to attacker

Mitigations (addressed in profile / OAuth Security BCP / OAuth 2.1):

- **Simple String Comparison ONLY for Redirect URI – no wildcards allowed**
- **Use PKCE**

OAuth Cross-Site Request Forgery (CSRF) Attack

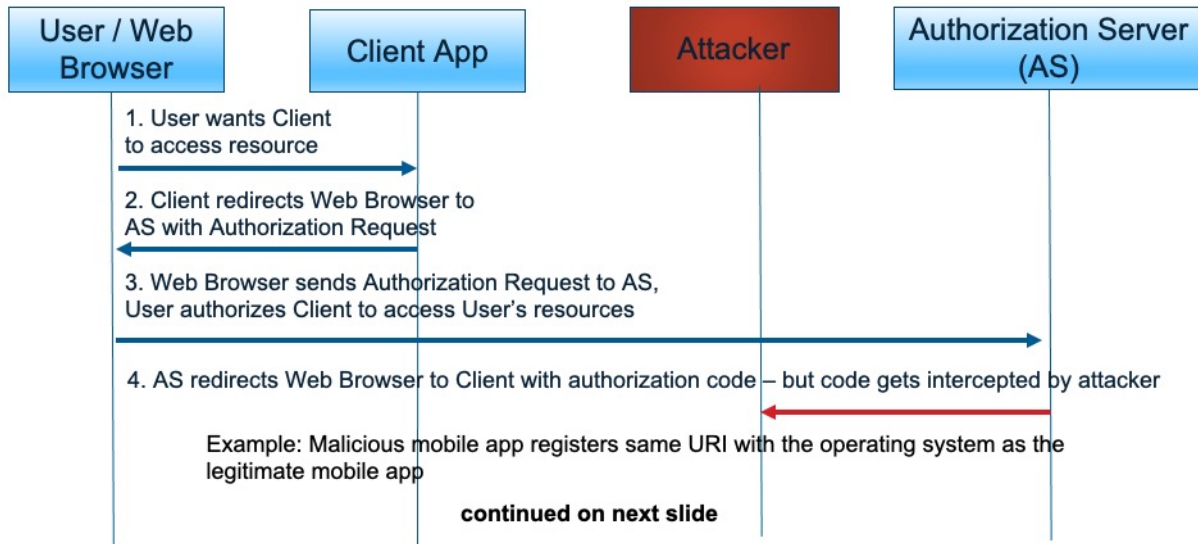


MITRE

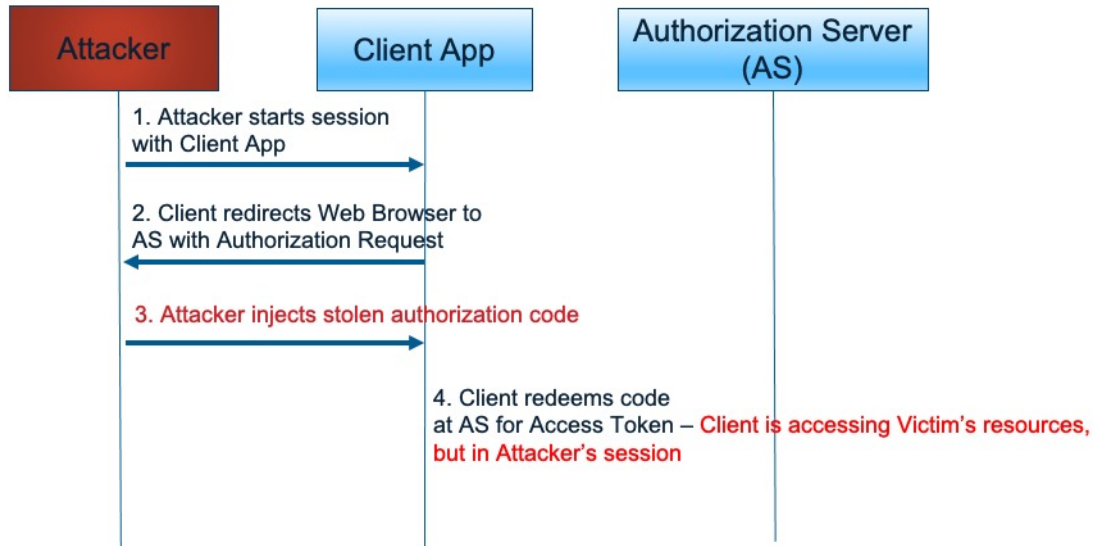
© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

31

OAuth Authorization Code Injection Attack



OAuth Authorization Code Injection Attack



MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

33

Authorization Code Injection Mitigations

Use Proof Key for Code Exchange (PKCE – IETF RFC 7636)

Authorization Request bound to client's session with user

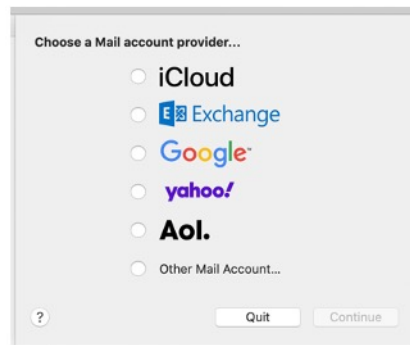
Only the same client – and same user session on the client – can redeem authorization code for an access token

Claimed URIs on Android, iOS, Windows, etc.

Create strong binding in operating system between URI and app
e.g., <https://oauth.facebook.com> belongs to genuine Facebook app
(bind through Android app signing certificate, iOS app package name)

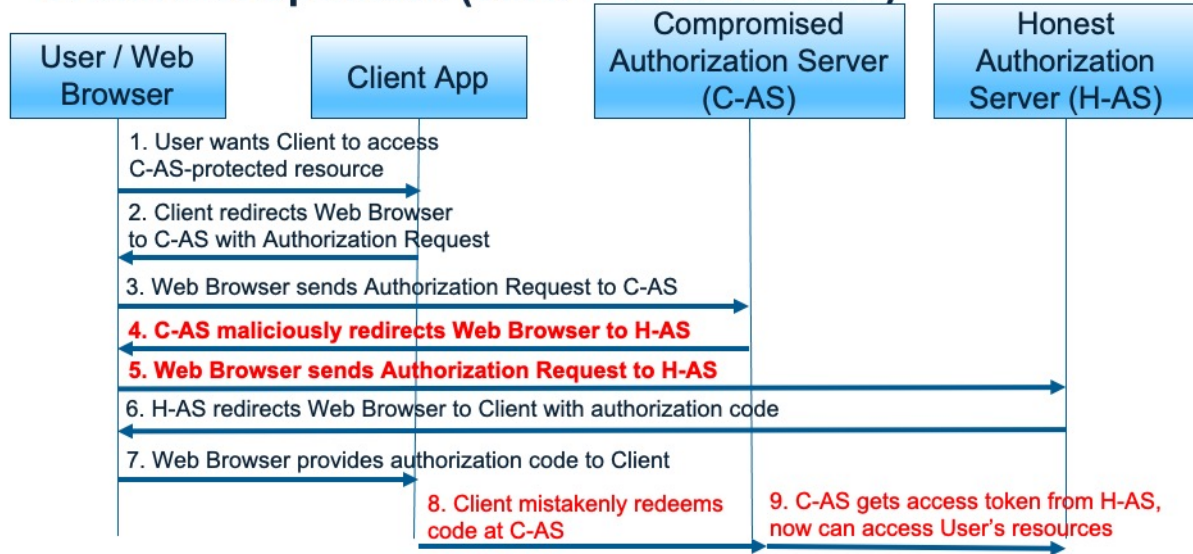
OAuth Mix-Up Attack

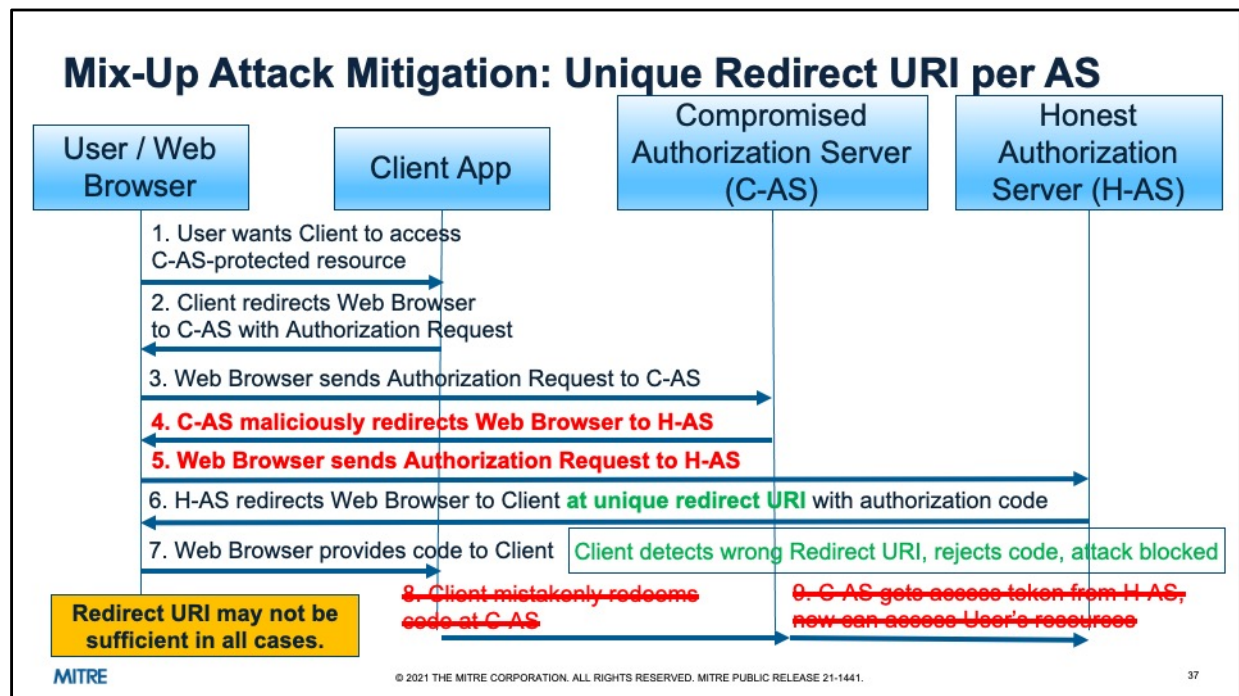
- **May occur if OAuth Client supports multiple Authorization Servers (AS)**
 - Compromised AS tricks Client, obtains access to user's resources at another AS



**IETF OAuth WG is aware of mix-up attacks and has proposed solutions
Will take time to get finalized and implemented**

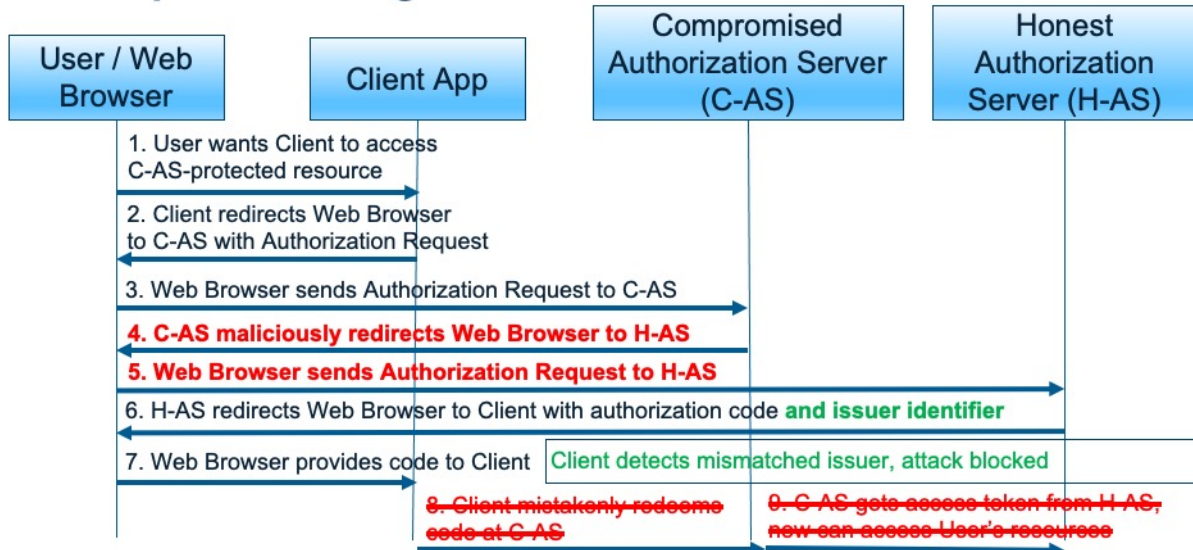
OAuth Mix-Up Attack (other variations exist)





- Redirect URI not always sufficient -- pointed out by Daniel Fett to IETF OAuth Working Group in October 2020
- <https://mailarchive.ietf.org/arch/msg/oauth/RjbSwFRmLsk0EgAY2Ter-nw66EY/>

Mix-Up Attack Mitigation: Add Issuer Identifier



MITRE

© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.

38

JWS Header Manipulation

- **“alg” field is in JSON Web Signature (JWS) header and is not protected**
- **JWS uses the term “signature” to encompass both asymmetric signatures and symmetric message authentication codes (MACs)**
- **Attacker changes asymmetric signature to symmetric “signature”**
 - Example: Change “RS256” (RSA with SHA-256) to “HS256” (HMAC with SHA-256)
- **Poorly implemented token recipient now treats RSA public key as an HMAC key**
 - Now attacker can use RSA public key to easily forge “signatures”
- **Or just change alg to “none”**

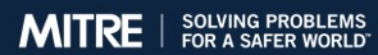
- **Mitigation: Test JWS libraries, ensure they are invoked properly**

JWS Header Manipulation

- **JWS optionally contains a “jwk” field containing the raw public key to be used to verify its own signature**
 - Bad idea that encourages developers to do the wrong thing
 - <https://www.ietf.org/mail-archive/web/jose/current/msg01674.html>
 - <https://www.ietf.org/mail-archive/web/jose/current/msg05589.html>
- **Attacker puts their own public key in the “jwk” field... naïve implementation uses public key to verify the JWS contents...**
- **Example: CVE-2018-0114 (Cisco node-jose open source library)**
- **Mitigation: Test JWS libraries, ensure they are invoked properly**

For more information

OAuthOIDCProfiles@groups.mitre.org



© 2021 THE MITRE CORPORATION. ALL RIGHTS RESERVED. MITRE PUBLIC RELEASE 21-1441.