Analyzing Privacy Implications and Security Vulnerabilities in Single Sign-On Systems: A Case Study on OpenID Connect

Mohammed Al Shabi[1], Rashiq Rafiq Marie[2]
Department of Management Information System, Taibah University, Madinah, Saudi Arabia[1]
Department of Information System, Taibah University, Madinah, Saudi Arabia [2]

***Abstract*—Single Sign-On (SSO) systems have gained popularity for simplifying the login process, enabling users to authenticate through a single identity provider (IDP). However, their widespread adoption raises concerns regarding user privacy, as IDPs like Google or Facebook can accumulate extensive data on user web behavior. This presents a significant challenge for privacy-conscious users seeking to restrict disclosure of their online activities to third-party entities. This paper presents a comprehensive study focused on the OpenID Connect protocol, a widely utilized SSO standard. Our analysis delves into the protocol's operation, identifying security flaws and vulnerabilities across its various stages. Additionally, we systematically examine the privacy implications associated with user access to SSO systems. We offer a detailed account of how easily user information can be accessed, shedding light on potential risks. The findings underscore the imperative to address privacy vulnerabilities within SSO infrastructures. We advocate for proactive measures to enhance system security and safeguard user privacy effectively. By identifying weaknesses in the OpenID Connect protocol and its implementations, stakeholders can implement targeted strategies to mitigate risks and ensure the protection of user data. This research aims to foster a more secure and privacy-respecting environment within the evolving landscape of SSO systems.**

Keywords*—*Single Sign-On; OpenID connect protocol; vulnerabilities; privacy; third-party

I. Introduction

Single Sign-On (SSO) systems streamline user access to various online services by consolidating credentials, thus eliminating the need for multiple usernames and passwords. However, this convenience often conflicts with user privacy, as identity providers (IDPs) involved in the SSO process can track and collect user data across platforms, potentially sharing it with third-party organizations for targeted advertising or profiling. Notably, prominent IDPs like Google and Facebook are known to leverage such data, raising significant concerns regarding user privacy and control over personal information [1] [2] . In light of these challenges, understanding the security and privacy implications of SSO systems becomes paramount.

OpenID Connect emerges as one of the most prevalent SSO protocols, offering a standardized framework for authentication and authorization across diverse websites and applications. This research endeavors to comprehensively assess the efficacy and potential vulnerabilities of OpenID Connect through a large-scale practical study. By dissecting its operational stages, we aim to identify security weaknesses and risks inherent in the protocol. The primary objective of this study is to evaluate the privacy implications of user access to SSO systems, along with the potential exposure of user information to IDPs and third-party entities. Through systematic analysis, we scrutinize the accessibility of user data within the SSO framework, illuminating the extent to which online user behavior can be monitored and exploited.

To underscore the real-world impact of these privacy concerns, we conduct a Mix-up attack on the OpenID Connect protocol using a local ASP.net-based API. The success of this attack in obtaining ID tokens, subsequently utilized to construct Access tokens for unauthorized resource access, underscores the vulnerability of SSO systems. Moreover, we quantify the attack's efficiency on online platforms by measuring the number of tokens accessed during execution.

This research endeavors to contribute significantly to the understanding of privacy implications and security vulnerabilities inherent in SSO systems. By shedding light on these issues, we aim to inform the development of more robust and privacy preserving SSO solutions. Ultimately, our findings advocate for empowering users to retain control over their personal data and limit disclosure to third-party organizations. The rest of the paper is organized as follows. Section II introduces the related works. Next, in Section III, the Open Connect Protocol is described, followed by Section IV which presents the vulnerabilities in Open ID connect. Section V states the vulnerabilities in OIDC. The implementation of a mix-up attack using ASP.net is described in Section VI. Finally, Section VII concludes the paper's work and findings.

II. Related Work

The security and privacy issues of Single Sign-On (SSO) systems and their protocols have been investigated by several studies. These studies have enhanced the understanding of the challenges and risks involved in the use of SSO systems. Some of the key research in this area is summarized as follows: The work in study [3] examines the tracking capabilities of third-party entities on the web and explores potential defences against such tracking. It sheds light on the privacy risks associated with SSO systems and the information that identity providers (IDPs) can gather about user behavior. The study in[4] investigates the security vulnerabilities of web-based password managers, which are often integrated with SSO systems. It analyzes potential attacks and risks to

user privacy when relying on SSO for password management and authentication.

The research in study [5] explores the trade-off between user control and usability in social networking platforms that utilize SSO. It discusses privacy-preserving mechanisms and approaches to mitigate the risks associated with sharing personal data through SSO systems. The study in [6] investigates the vulnerabilities and attacks related to account hijacking and session management in SSO systems. It provides insights into the security risks associated with SSO protocols and highlights the importance of robust authentication and session handling mechanisms. The survey in [7] provides an overview of the privacy challenges in Single Sign-On and explores potential solutions. It discusses various aspects of SSO privacy, including information leakage, tracking risks, and user consent. The research in [8] conducts a comparative analysis of security properties in different SSO protocols, including SAML, OAuth, and OpenID Connect. It identifies vulnerabilities and discusses security considerations in these protocols. This empirical [9] investigation focuses on the security and privacy aspects of OpenID Connect. It analyzes potential threats and vulnerabilities in the protocol and provides insights into its effectiveness in protecting user privacy. The literature review in study [10] examines various SSO protocols and their security and privacy characteristics. It identifies common vulnerabilities, threats, and mitigation strategies present in the literature. This systematic in [11] review and meta-analysis analyze the security and privacy aspects of SSO systems. It synthesizes findings from multiple studies and provides an overview of the state-of-the-art research in the field. This study in [12] presents a case study focusing on the privacy-sensitive features in a web authentication system. The researchers analyze and evaluate the privacy implications of various features and mechanisms used for web authentication. The study sheds light on the potential privacy risks and challenges that users may face during the authentication process and provides insights into the design of privacy-preserving authentication systems.

TABLE I. KEY RESEARCH WORKS

Study	Key contribution	Focus
Roesner et al. [3]	Examined tracking capabilities of third-party entities on the web.	Privacy risks and information gathering by identity providers (IDPs) in SSO systems.
Chia [4]	Investigated security vulnerabilities of web-based password managers integrated with SSO.	Attacks and privacy risks in SSO systems for password management and authentication.
Rahman [5]	Explored the trade-off between user control and usability in SSO-based social networking.	Privacy-preserving mechanisms and risk mitigation for sharing personal data via SSO systems.
Nergiz [6]	Investigated vulnerabilities and attacks related to account hijacking and session management.	Security risks in SSO protocols, emphasizing robust authentication and session handling.
Yang et al. [7]	Provided an overview of privacy challenges in Single Sign-On and potential solutions.	SSO privacy aspects, including information leakage, tracking risks, and user consent.
Fett et al. [8]	Conducted a comparative analysis of security properties in different SSO protocols.	Vulnerabilities and security considerations in SAML, OAuth, and OpenID Connect protocols.
W. Li and C. J. Mitchell [9]	Empirical investigation of security and privacy aspects of OpenID Connect.	Analysis of threats and vulnerabilities in the protocol and its effectiveness in protecting user privacy.
Ahmad et al. [10]	Literature review of various SSO protocols and their security and privacy characteristics.	Identification of common vulnerabilities, threats, and mitigation strategies from literature.
Zuo et al. [11]	Systematic review and meta-analysis of security and privacy aspects of SSO systems.	Synthesis of findings from multiple studies to provide an overview of state-of-the-art research in the field.
Li et al. [12]	Evaluated privacy implications of identity federation in SSO systems.	Privacy risks and concerns associated with identity federation in SSO deployments.
Wang et al. [13]	Analyzed authentication vulnerabilities in SSO for mobile apps.	Security weaknesses and risks in SSO implementations for mobile applications.
Meland et al. [14]	Investigated the privacy implications of attribute sharing in SSO systems.	Risks associated with sharing user attributes among multiple service providers through SSO.
Paul et al. [15]	Explored privacy and security challenges in the context of healthcare SSO systems.	Addressing privacy concerns and enhancing security for SSO implementations in healthcare environments.

This research in study [13] provides a comprehensive study of OAuth security issues specifically related to Android apps. The authors analyze the implementation of OAuth in various Android applications to identify potential security vulnerabilities and weaknesses. The study uncovers security risks and potential attacks that could compromise the security and privacy of user data when using OAuth-based authentication in Android apps. This study in [14] investigates the privacy aspects of attribute sharing in Single Sign-On (SSO) solutions. The researchers examine the process of attribute sharing among multiple service providers in SSO systems and analyze the privacy risks associated with this sharing. The study aims to enhance the understanding of the potential privacy concerns and challenges in SSO deployments and provides insights into protecting user privacy in attribute sharing scenarios.

This systematic review in [15] focuses on Single Sign-On (SSO) security specifically in the context of healthcare. The researchers review and analyze existing literature on SSO security, with a specific focus on healthcare environments. The study identifies security challenges and vulnerabilities related to SSO implementations in healthcare settings and provides recommendations for enhancing security and privacy in healthcare SSO systems. The review contributes to a better understanding of SSO security concerns and implications in healthcare applications. Table I summarizes the key research works related to privacy implications and security vulnerabilities associated with Single Sign-On (SSO) systems.

III. OPEN CONNECT PROTOCOL

OpenID Connect is an industry-standard protocol used for authentication and authorization in Single Sign-On (SSO)

systems. It is built on top of the OAuth 2.0 framework and provides a standardized way for users to log in to multiple websites or applications using a single set of credentials [16]. The primary goal of OpenID Connect is to enable identity federation, allowing users to authenticate with an identity provider (IDP) and then use that authentication to access various relying parties (RPs) without needing separate credentials for each RP. The IDP is responsible for verifying the user's identity and providing the necessary authentication tokens. Fig. 1 depicts the relationship among the components of OpenID.

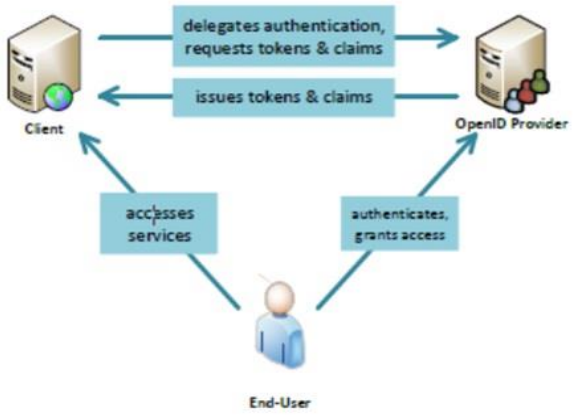

Fig. 1. The relationship between the components in the protocol.

A. Brief overview of how OpenID Connect works

- User Initiation: When a user attempts to access a website or an application (RP) that supports OpenID Connect, they are redirected to the IDP's authentication endpoint.
- Authentication: The user is prompted to enter their credentials (e.g., username and password) at the IDP's login page. The IDP authenticates the user and generates an ID token.
- ID Token Exchange: After successful authentication, the IDP issues an ID token to the RP. This ID token contains information about the user and the authentication event, such as the user's unique identifier and any requested user claims.
- Token Validation: The RP validates the ID token to ensure its integrity and authenticity. It checks the token's signature, expiration, and the IDP's issuer information to verify its validity.
- User Authorization: Once the RP has validated the ID token, it can authorize the user's access to its resources based on the user's identity and any additional authorization scopes or claims provided.

OpenID Connect also supports optional features such as UserInfo endpoint, which allows RPs to retrieve additional user profile information from the IDP, and the use of refresh tokens for obtaining new access tokens without re-authentication. By leveraging OpenID Connect, SSO systems can offer a seamless and secure user experience, as users only need to authenticate once with their IDP and can then access multiple applications without the need for separate logins. The protocol facilitates interoperability among different identity providers and relying parties, providing a standardized framework for SSO implementation. Fig. 2 illustrates the implementation phases in the OpenID Connect protocol.

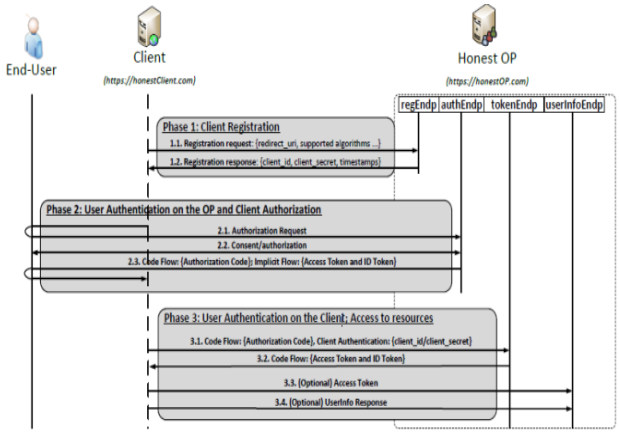

Fig. 2. Implementation phases in the OpenID Connect protocol.

The stages of the OpenID Connect protocol work:

1) Stage one: Dynamic registration and discovery in Fig. 3 shows the initial registration stage more accurately, In the beginning, the user presents his identity (for example, alice@honestOP.com) to the customer to obtain services. To authenticate the user, the client needs to discover the IDP which controls the identity of the alias.

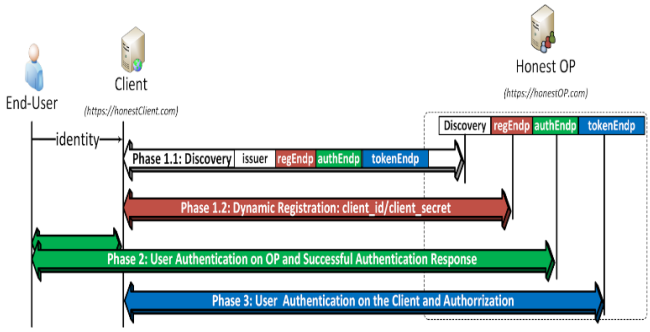

Fig. 3. OpenID Connect dynamic registration phase.

The first stage is divided into two steps:

- Step 1 (Discovery): The client sends a request to the Discovery endpoint and returns OP's configuration information including the locations of the endpoints.

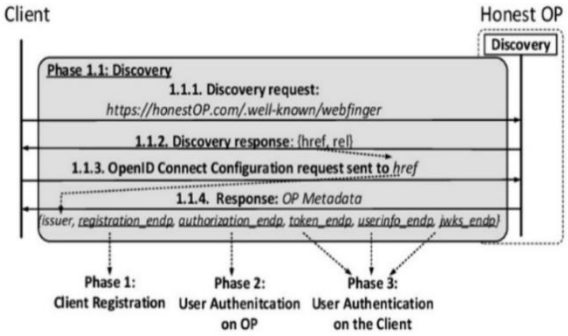

Fig. 4. Detailed overview of the discovery phase in OIDC.

Fig. 4 shows the details of the discovery phase in which the metadata received by the client appears and its impact on the protocol phases. Note that this data contains all information related to the protocol (endpoints, signature, and encryption algorithms - messages - public keys of the protocol).

- Step 2 (dynamic registration): The client automatically registers with the Identity Provider (OP) where it sends its own address (for example http://client.com) to the registration endpoint address, so the OP responds and sends the pair (/client_id client_secret), which are secret codes between the client and the Identity Provider (OP) representing Client credentials.

2) Stage two: Authenticating In this phase the user is authenticated at the identity provider (User: Authentication on the OP). The client directs the unauthenticated user to an authorization endpoint (the user is directed to an address on which the client ID is pre-registered). The end user is authenticated to the OP using his or her credentials. The OP sends an authorization code that includes an (Access token, ID token) this code is an intermediary between the client and the user through which the client can access specific resources for the user and verify the identity of the end user.

3) Stage three: User Authentication stage (on the OP - ID and Access Token): After the client receives the code at the end of the second stage, it sends it to the token endpoint and sends its credentials from the first stage (client_id and client_secret), then the identity provider responds and sends it the (access token, ID token), and then the client Verifies the tokens and performs final user authentication.

- ID Token content

```
Header: {"alg": "HS256"}
Body: {
      iss": "http://openidConnect
Provider.com/", "sub": "user1",
      "exp": 1444148908,
      "iat": 1444148308,
      "nonce": "40c6b33b9a2e",
      "aud": "http://client.com/",}
Signature: AF45JF93LKD76D...
```

Fig. 5. An example of an ID token.

An id token is a secret token that contains information (claims) about the end user's identity and structure. Its data is a JWT (JSON Web Token). Fig. 5 shows an example of an id token [17].

We note that the id token consists of three parts:

First: The header contains information that includes the encryption algorithm used.

Second: The body contains the information needed to authenticate the end user, including:

End User ID: It consists of two parts:

- Issuer (Iss): To know the identity provider.
- sub (Subject o): To know the identity of the user.
- The timestamp (iat) and expired (exp) define the time period for the token to be produced and to expire.
- Nonce: A random string sent by the client during the authentication request used to mitigate attacks.
- Audience (aud): Determines which customers the identity token belongs to.

Third: Signature provides the reliability of the id token.

IV. OpenID Connect Vulnerabilities

One notable vulnerability in the OpenID Connect (OIDC) protocol is the introduction of dynamic registration and discovery, a phase that was absent in previous protocols. This phase became a significant weakness in the OIDC protocol, as attackers were able to manipulate the information, particularly endpoint addresses, exchanged during the discovery phase. Attackers could substitute legitimate addresses with their own, enabling them to intercept and manipulate information exchanged between clients and the attacker throughout the various phases of the protocol. The layered structure of the OIDC protocol further facilitated attackers in intervening between any two of the three primary phases. Several previous studies have explored and exploited these vulnerabilities, leading to the development of new attacks targeting the OIDC protocol and the creation of tools for analyzing and testing its confidentiality.

A. OpenID Connect Previous Studies

Security vulnerabilities in single sign-on protocols have been examined, analyzed, and identified in a number of previous studies. Here are some of the most significant contributions made by these studies; in addition, Table II summarizes the principal contributions made by the mentioned studies.

In study [17] the researchers conducted a comprehensive examination of the security characteristics of the Google OIDC protocol. They examined a group of clients and applied a class of attacks to obtain user codes, enabling impersonation and unauthorized access to clients. Additionally, they provided valuable insights and future recommendations for both Service Providers (RPs) and Identity Providers (OPs) to enhance security in future systems. Researchers in [18] developed a tool to test attacks on the OpenID Connect (OIDC) Access Protocol. This tool encompasses advanced attacks, including malicious endpoint attacks, contributing to a better understanding of the protocol's message flow.

In study [19] the researchers conducted an in-depth analysis of the OIDC protocol's dynamic registration and discovery feature. They identified a new type of attack named "Malicious Endpoints" that exploits information exchanged between protocol parties, posing risks to user privacy. In researcher [20] the researchers analyzed known attacks on the OIDC protocol and classified them into two categories: single-stage attacks, targeting one stage, and two-stage attacks, relying on multiple stages. They also proposed measures to

enhance protocol confidentiality and identified security vulnerabilities. Researchers in [21] provided a thorough and formal security analysis of the OpenID Connect protocol. They developed a model of OpenID Connect, utilizing secret features to mitigate known attacks, and put forth security guidelines to bolster the overall security posture of the protocol. The authors of [22], studied the security flaws of OAuth 2.0 in Android apps, focusing on the implementation errors that could lead to breaches. They also discussed the OAuth security challenges specific to the Android platform. Research in [23] examined the security misconfigurations in mobile OAuth implementations, using real-world apps as examples. They uncovered common mistakes that could affect the security of mobile systems based on OAuth. In paper [24], performed a comprehensive security assessment of OAuth 2.0, which is the authorization framework for OpenID Connect. They evaluated different OAuth implementations and detected vulnerabilities, and they suggested recommendations for enhancing the security of systems based on OAuth. The researchers in [25], investigated the security of OAuth, which is the foundation for OpenID Connect. They analyzed the weaknesses and vulnerabilities in OAuth's authentication and authorization mechanisms, and they highlighted the potential risks associated with OAuth implementations.

This paper in [26] presents a Systematization of Knowledge (SoK) on OAuth 2.0 and explores the current vulnerabilities, limitations, and ongoing efforts to improve its security. The study identifies various threats and vulnerabilities related to the authentication and authorization mechanisms of OAuth 2.0. It also discusses potential solutions and ongoing research to enhance the protocol's security in protecting user privacy and data. This research [27]provides an in-depth formal security analysis of the OpenID Connect protocol. By applying formal methods, the study examines the security properties and potential vulnerabilities of OpenID Connect. The authors develop a model of the protocol and provide security guidelines to mitigate. This survey article[28] presents a comprehensive comparison of security modelling approaches for various Single Sign-On (SSO) protocols, including OpenID Connect. The study reviews the strengths, weaknesses, and challenges in the security modelling of SSO protocols, providing insights into the security aspects of OpenID Connect and other SSO protocols known attacks and enhance the overall security posture of OpenID Connect.

This study in [29] investigates the design and security considerations of a Mobile Single Sign-On (SSO) system based on OpenID Connect. The research examines the integration of OpenID Connect for mobile applications, focusing on the design aspects and security measures necessary to ensure a secure and user-friendly SSO experience on mobile platforms. This empirical [30]investigation delves into the security, privacy, and usability aspects of the OpenID Connect protocol. The research assesses potential threats and vulnerabilities in OpenID Connect's implementation and analyzes its effectiveness in safeguarding user privacy and data. Additionally, the study evaluates the usability of OpenID Connect in real-world scenarios. This survey paper [31] provides a comprehensive examination of OAuth-based Single Sign-On (SSO) protocols, including OpenID Connect. The study reviews different SSO protocols and focuses on OAuth's role as the foundation for SSO mechanisms. The paper discusses the strengths and weaknesses of OAuth-based SSO and highlights areas for further improvement. This research [32] conducts a security analysis of the OAuth 2.0 framework in the context of mobile applications. The study identifies potential vulnerabilities and threats associated with OAuth 2.0 when utilized in mobile environments. The paper discusses security considerations and suggests measures to enhance the protection of user data and privacy in OAuth-based mobile applications.

TABLE II. SUMMARIZING THE KEY CONTIBUTIONS AND FOCUS OF THE MENTIONED STUDIES RELATED TO OPENID CONNECT AND OAUTH 2.0 SECURITY

Study	Key Contributions	Focus
Li et al. [17]	- Examined Google OIDC protocol security characteristics. - Conducted attacks to obtain user codes and impersonate clients. - Provided future recommendations for RPs and OPs.	- Security analysis of Google OIDC. - Attack scenarios on OIDC clients. - Recommendations for enhancing security.
Mladenov et al. [18]	- Developed a tool for testing OIDC Access Protocol attacks. - Enhanced understanding of the protocol's message flow.	- Advanced attacks, including malicious endpoint attacks in OIDC.
Mainka et al. [19]	- Conducted in-depth analysis of OIDC's dynamic registration and discovery feature. - Identified "Malicious Endpoints" attack and its risks.	- Analysis of OIDC's dynamic registration and discovery feature.
Navas et al. [20]	- Analyzed known attacks on the OIDC protocol and classified them into single-stage and two-stage attacks. - Proposed measures to enhance protocol confidentiality and identified security vulnerabilities.	- Categorized attacks into single-stage and two-stage attacks.
Fett et al. [21]	- Provided thorough and formal security analysis of the OpenID Connect protocol. - Developed a model of OpenID Connect and security guidelines for mitigation.	- Formal security analysis of OpenID Connect protocol.
Maqbool et al. [22]	- Studied the security flaws of OAuth 2.0 in Android apps. - Focused on implementation errors and their impacts.	- Security assessment of OAuth 2.0 in Android apps. - OAuth security challenges on the Android platform.
A. Kountouras and G. Frantzeskou [23]	- Examined security misconfigurations in mobile OAuth implementations using real-world apps. - Identified common mistakes affecting mobile OAuth security.	- Examination of real-world mobile OAuth implementations.
J. Richer and A. Sanso [24]	- Performed comprehensive security assessment of OAuth 2.0. - Evaluated vulnerabilities in various OAuth implementations. - Suggested recommendations for enhancing OAuth 2.0 security.	- Evaluation of OAuth 2.0 vulnerabilities and security measures.
B. Braithwaite and A. Doupé [25]	- Investigated security vulnerabilities in OAuth's authentication and authorization mechanisms. - Highlighted potential risks associated with OAuth implementations.	- Analysis of OAuth's weaknesses and vulnerabilities.
Pereira et al. [26]	- Systematization of Knowledge (SoK) on OAuth 2.0 security. - Identification of threats and vulnerabilities related to OAuth's authentication and authorization mechanisms. - Discussion of potential solutions to improve OAuth's security.	- Identification of threats and vulnerabilities in OAuth 2.0. - Ongoing research to enhance OAuth's security and protect user privacy.

Küsters and Kifayat [27]	- Provided in-depth formal security analysis of the OpenID Connect protocol. - Developed a model of OpenID Connect and provided security guidelines.	- Formal security analysis of OpenID Connect protocol.
Sanchez-Aguilar et al. [28]	- Comprehensive comparison of security modeling approaches for various Single Sign-On (SSO) protocols, including OpenID Connect. - Review of strengths, weaknesses, and challenges in security modeling of SSO protocols.	- Review of security modeling approaches for SSO protocols. - Insights into the security aspects of OpenID Connect and other SSO protocols.
Vetrivelan et al. [29]	- Investigated design and security considerations of a Mobile Single Sign-On (SSO) system based on OpenID Connect. - Focused on secure integration of OpenID Connect for mobile applications.	- Design aspects and security measures for secure Mobile SSO with OpenID Connect.
Alshehri et al. [30]	- Delved into the security, privacy, and usability aspects of the OpenID Connect protocol. - Assessed threats and vulnerabilities in OpenID Connect's implementation. - Evaluated the effectiveness of OpenID Connect in safeguarding user privacy and data.	- Security, privacy, and usability evaluation of OpenID Connect.
Sun et al. [31]	- Provided a comprehensive examination of OAuth-based Single Sign-On (SSO) protocols, including OpenID Connect. - Focused on OAuth's role as the foundation for SSO mechanisms. - Discussed strengths and weaknesses of OAuth-based SSO.	- Review of OAuth-based SSO protocols. - Identification of strengths and weaknesses.
Khan and Shafiq [32]	- Conducted a security analysis of the OAuth 2.0 framework in the context of mobile applications. - Identified vulnerabilities and threats in OAuth 2.0 for mobile environments. - Suggested measures to enhance user data and privacy protection in OAuth-based mobile applications.	- Security analysis of OAuth 2.0 in mobile applications. - Measures to enhance OAuth-based mobile app security.

V. OIDC Vulnerabilities

1) Phishing: Phishing attacks in OIDC can take two forms:

a) Spoofed OP page: Attackers can redirect users to a spoofed Identity Provider (OP) page where they are deceived into entering their OP credentials[20].

b) Realm spoofing: Service Providers (RPs) can craft authentication requests with an OpenID realm parameter set to a trusted domain but redirect the user back to their own page without proper verification. The user's OP falsely assures them that they are logging into the trusted domain, while they are redirected to the RP.

2) Session related attacks: OIDC allows multiple active authentication sessions, providing more opportunities for malicious sites to exploit vulnerabilities in both OPs and RPs[33]. Specific issues include:

a) Session swapping: Lack of a mechanism to associate an OIDC session with the user's browser allows attackers to configure an attacker-authenticated session in the RP. This can lead to unauthorized access and disclosure of sensitive information.

b) Cross-Site Request Forgery (CSRF): Logged-in users may be susceptible to CSRF attacks targeting the OP or other RP sites.

c) Cross-Site Scripting (CSS): Logged-in users may be vulnerable to XSS attacks against the OP or other RP sites.

3) Data privacy: OpenID Connect service providers have visibility into every site the user logs into using their credentials. This centralized nature allows malicious OPs to track user activity on the internet.

4) Risk centralization: Hacker's target IdP accounts as they provide access to multiple sites. If an identity provider lacks escalated authentication options, the user's security may be compromised.

5) Weak background security: Some OPs may rely solely on email account recovery, which is inadequate for strong background security. Account recovery mechanisms should employ stronger authentication methods.

6) Cross-identifier relationship: When users employ the same OpenID across different RP sites, these sites can link user information or activity. Independent logins in different RPs mitigate this issue.

7) Shared ID strings: The usability challenge of entering ID strings increases when different launches employ different methods.

This can lead to users inadvertently entering the same ID in multiple places, allowing RPs to establish relationships between them.

VI. Implementing A Mix-Up Attack Using Asp.Net

A. Attack Assumption

- The RP (Service Provider) is configured to connect to multiple OAuth Providers (Ids), including one controlled by the attacker (AldP) and another harmless one (HIP).
- The RP uses the same redirect_uri for multiple identity providers and relies on the 'state' parameter to determine the origin of the response.
- The attacker's AldP has control over the authentication flow and response sent to the RP.

B. Attack Stages

Stage 1: The attacker obtains the victim's token from HldP.

Fig. 6 shows the steps of this stage [9], he following steps illustrate the attack flow:

1) The End User clicks the "Login with HldP" button on an RP page.

2) The attacker communicates with the RP and sends a login request to the RP, pretending to be the AldP (attacker-controlled Identity Provider). As a result, the attacker receives a redirect response at the AldP's authorization endpoint.

3) The attacker redirects the response to the browser, intending it to be transmitted to the conversion endpoint of HldP (harmless Identity Provider). However, during this process, the attacker uses the status value associated with the authorization request for the AldP that was received in step 2.

4) The end user interacts with HldP and clicks "Agree" to authenticate and authorize the request.

5) HldP returns the end user to the redirect_uri of the RP, along with the associated token.

6) The RP receives the token and evaluates the status value, determining that the authorization response is from the AldP (attacker's-controlled Identity Provider).

7) The RP mistakenly sends the code and tokens to the AldP's token endpoint and API endpoints, allowing the HldP tokens to be delivered to the attacker.

Stage 2: Using the victim's code V.

In this step, the RP continues to interact with the attacker, assuming the attacker is the victim. If the RP provides data or allows modifications based on the access token, the attacker gains access to the victim's resources and can potentially manipulate them, Fig. 7 shows this stage.

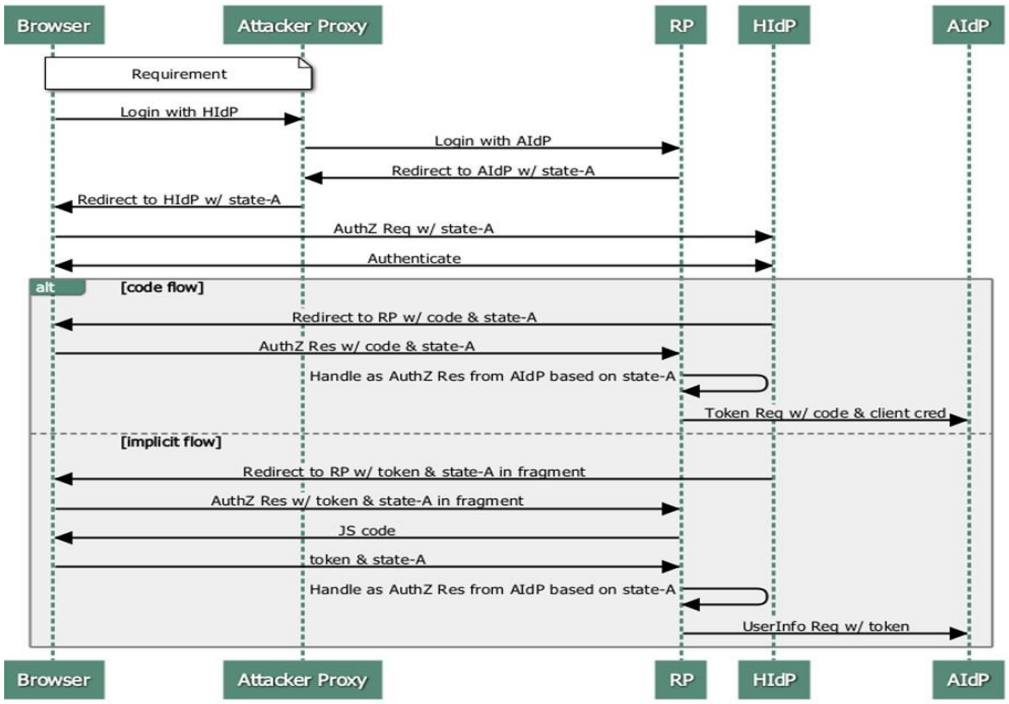

Fig. 6. Mix-up attack steps.

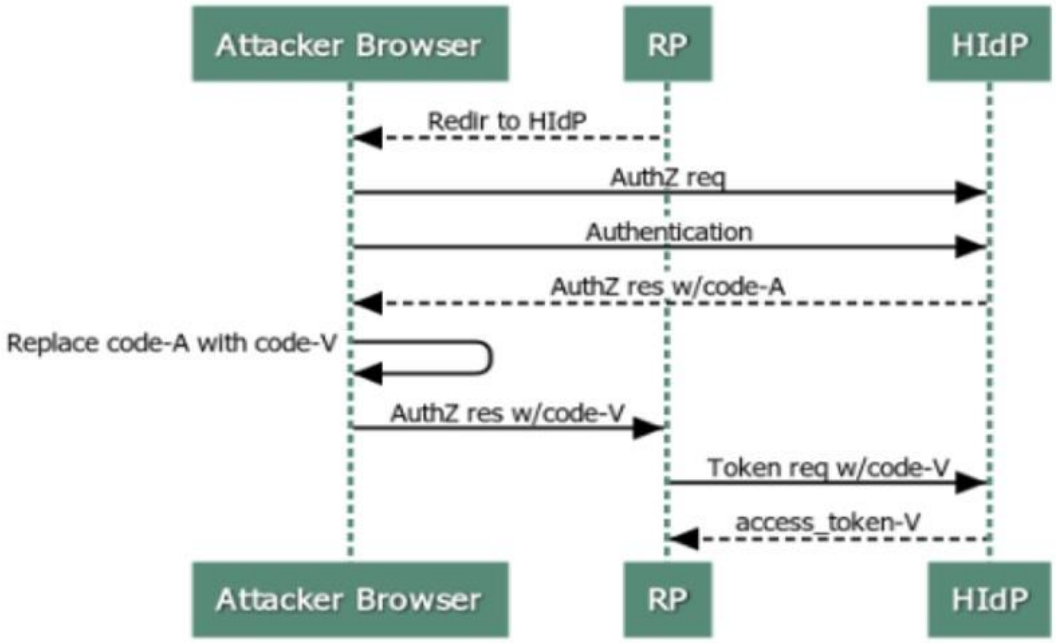

Fig. 7. Victim code used by the attacker.

C. Execute Attack on API

The provided excerpt describes the execution of the attack on the API and the consequences of its success. Here's an explanation of the details mentioned:

1) Successful Attack and Obtaining User Information:

a) When the attack is successful, the attacker receives information in JSON format.

b) The attack allows the attacker to access user values that were previously protected.

c) The obtained user values are stored in two encrypted strings or variables within the API, referred to as value 1 and value 2. Fig. 8 illustrates the protected user token values prior to the attack.

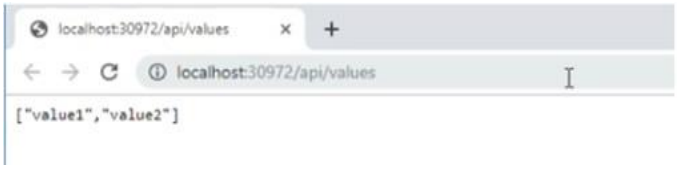

Fig. 8. Protected user token values prior to attack.

2) Obtaining user ID token:

a) After executing the attack, the attacker obtains the user's ID token.

b) The ID token is typically encrypted using JWT (JSON Web Token) and needs to be decrypted to extract its contents. Fig. 9 illustrates the ID token values after the attack has been applied.

Fig. 9. The id token values after the attack has been applied.

3) Decrypting the ID token:

a) The attacker decrypts the ID token using JWT services.

b) Decrypting the token allows the attacker to access the user's tokens.

4) Accessing user resources:

a) Once the attacker has obtained the necessary tokens, they can use them to log in and gain access to the user's resources.

b) These by bypasses the protection mechanisms of the protocol, potentially resulting in the loss of user data and compromising their account.

D. Online Attack Implementation

The applied attack steals a set of user account tokens that are as in Table III. These codes play a critical role in interacting with the protocol's service provider, and their theft grants the attacker access to the victim's data. It is important to note that the success of the attack relies on the service provider enabling redirection for the user account, which is commonly known as a redirect attack. Upon executing the attack on a Gmail account, the codes in Fig. 10 were obtained, which illustrates the obtained tokens, revealing a successful attack where most of the mentioned tokens were acquired. However, it is important to note that certain tokens, such as EXP Idp tokens, were not obtained, resulting in a lack of information regarding the specific hack.

TABLE III. USER ID TOKEN

Token	Data Format	Meaning
Alg	String	Indicates the algorithm that was used to sign the token.
Kid	String	Sets the fingerprint of the public key that can be used to validate the signature of this token.
Iss	Source URL string	Defines the source or authorization server that generates and returns the token.
Idp	"String", usually the STS URL	Recording the identity provider that authenticated and returns the token subject.
Sub	String	The subject prompt value is immutable and cannot be modified or reused. It serves as a binary identifier that is unique to a particular application identifier. When a user logs into multiple applications with distinct customer IDs, each app will receive a different subject prompt value. The desirability of this behavior depends on your specific privacy requirements and system structure.
Hasgroups	Bool	If it exists, it is always true, indicating that the user is in at least one group. It is used in place of the collections claim for JWTS in implicit grant flows if the full collections claim would extend the URI part beyond the URL length limits (currently 6 or more groups).
Exp	String	Indicates when the session with the identity provider expired.
Website	String	Referring to the identity provider type.
Aud	String	The name of the site where the protocol is being used.

In the subsequent Table IV, we will delve into the attack on multiple platforms utilizing the protocol, providing a comparison between these platforms based on an essential criterion: redirection to the victim's page from which the codes were obtained. Redirection refers to the ability of the service provider to access the user's account and gain complete control over it through redirection techniques. Notably, prominent companies like Microsoft and Google have implemented measures to protect their users from such redirects, thus preventing unauthorized access to accounts. By applying the attack on the platforms listed in Table IV below, we can gauge the success of the attack by the number of tokens obtained. In this instance, we were able to acquire nine access tokens, indicating a 100% success rate for the attack.

TABLE IV. MIX-UP ATTACK PERCENTAGES ON POPULAR PLATFORMS

Platform	Token Access Rate	Redirect by the Service Provider
Gmail	7/9=77.7%	impossible
Facebook	5/9=55.5%	impossible
Yahoo	8/9=88.8%	possible
Hotmail	9/9=100.00%	possible
Outlook	4/9=0.44%	impossible

It is important to highlight the possibility of redirection in this context. Fig. 10 depicts the user tokens taken after applying the MIX-Up attack on the Gmail platform.

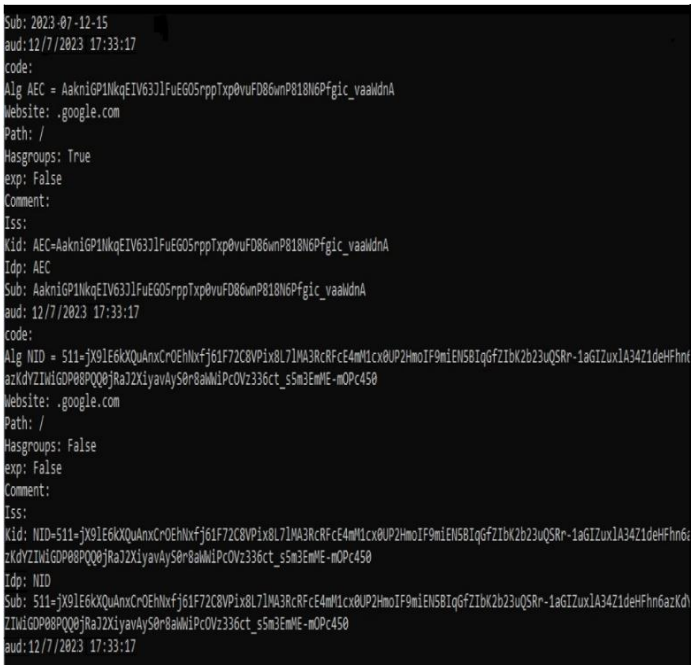

Fig. 10. User tokens after applying the MIX-Up attack on the gmail platform.

From the previous table, it is evident that the attack was successful across various popular platforms, yielding favorable success rates. However, variations in the verification rates of the attack can be observed due to factors associated with the response behaviour of the identity providers. Some servers permit code injection during the session, whereas others do not allow such manipulations. Additionally, the results differ based on the number of tokens accessed, which can vary depending on the specific protocol version implemented within each platform.

VII. CONCLUSION

In this research, we have examined the OpenID protocol and its information exchange mechanism, while also highlighting its significant security vulnerabilities and the execution of attacks. We have identified key design flaws within the protocol and successfully demonstrated a MIXup attack, resulting in the theft of user tokens and unauthorized access to user data in an offline attack scenario using the ASP environment. The attack was performed on accounts belonging to popular platforms, and the outcomes varied based on the characteristics of the session created by the identity provider.

It is worth noting that most renowned platforms have implemented measures to safeguard against redirect attacks by introducing browser-generated session values (routing fingerprints). They have also incorporated additional protection mechanisms, such as user confirmation via phone numbers or alternative email addresses. However, the accessed codes still present the possibility of conducting a redirect attack by deceiving the victim through a fraudulent page aimed at confirming the account and capturing return information within the user session. These findings underscore the importance of implementing enhanced protection measures in the future.

REFERENCES

[1] L. Smith, J., & Johnson, “The Privacy Implications of Single Sign-On Services,” in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 2019, pp. 1–14.

[2] A. Leung, R., & Datta, “Single Sign-On as Surveillance Infrastructure: Lessons from Google’s SSO Service,” in Proceedings on Privacy Enhancing Technologies, 2019, pp. 4–22.

[3] F. Roesner, T. Kohno, and D. Wetherall, “Detecting and Defending Against {Third-Party} Tracking on the Web,” in 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12), 2012, pp. 155–168.

[4] Z. Li, W. He, D. Akhawe, and D. Song, “The {Emperor’s} new password manager: Security analysis of web-based password managers,” in 23rd USENIX Security Symposium (USENIX Security 14), 2014, pp. 465–479.

[5] & H. Rahman, M. S., Kagal, L., “Rahman, M. S., Kagal, L., & Hendler, J. (2018). Sharing personal data while preserving privacy in social networking: Balancing user control and usability,” Comput. Secur., vol. 77, pp. 109–124, 2018.

[6] M. E. Nergiz, M. E., Mitchell, C. J., & Nergiz, “An empirical analysis of Single Sign-On Account Hijacking and session management on the web,” J. Inf. Secur. Appl., vol. 47, pp. 158–169, 2019.

[7] H. Yang, Z., Xing, L., & Chen, “Understanding and enhancing the privacy of Single Sign-On: A survey,” Comput. Commun., vol. 145, pp. 1–17, 2019.

[8] D. Fett, D., Küpser, A., & Schröder, “Security analysis of Single Sign-On protocols: Comparing SAML, OAuth, and OpenID Connect,” in In International Conference on Trust and Privacy in Digital Business, 2019, pp. 1–17.

[9] W. Li and C. J. Mitchell, "User Access Privacy in OAuth 2.0 and OpenID Connect," 2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), Genoa, Italy, 2020, pp. 664-6732, doi: 10.1109/EuroSPW51379.2020.00095

[10] N. Ahmad, F., Iqbal, M., Ahmad, J., & Alrajeh, “Security and privacy analysis of Single Sign-On protocols: A systematic literature review,” J. Inf. Secur. Appl., vol. 60, 2021.

[11] J. Zuo, C., Li, L., & Zeng, “Security and privacy analysis of Single Sign-On: A systematic review and meta-analysis,” Futur. Gener. Comput. Syst., vol. 119, pp. 103–116, 2021.

[12] L. F. Li, Y., Hong, J. I., & Cranor, “A case study of privacy-sensitive features in a web authentication system,” in In Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference, 2010, pp. 365–378.

[13] L. Wang, X., Lin, Z., Wang, X., Zhou, Y., & Xing, “A comprehensive study of OAuth security issues in Android apps,” in In Proceedings of the 33rd Annual Computer Security Applications Conference, 2017, pp. 312–325.

[14] M. G. Meland, P. H., Jensen, C. D., & Jaatun, “Privacy aspects of attribute sharing in single sign-on solutions,” in In International Conference on Availability, Reliability, and Security, 2017, pp. 57–74.

[15] P. J. Taylor, T. Dargahi, A. Dehghantanha, R. M. Parizi, and K.-K. R. Choo, “A systematic literature review of blockchain cyber security,” Digit. Commun. Networks, vol. 6, no. 2, pp. 147–156, 2020. https://doi.org/10.1016/j.dcan.2019.01.005

[16] C. Mainka, V. Mladenov, J. Schwenk and T. Wich, "SoK: Single Sign-On Security — An Evaluation of OpenID Connect," 2017 IEEE European Symposium on Security and Privacy (EuroS&P), Paris, France, 2017, pp. 251-266, doi: 10.1109/EuroSP.2017.32.

[17] W. Li and C. J. Mitchell, “Analysing the Security of Google’s implementation of OpenID Connect,” in DIMVA 2016: Proceedings of the 13th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment – vol. 9721J, pp.357–376, 2016, https://doi.org/10.1007/978-3-319-40667-1_18

[18] V. Mladenov, C. Mainka, and J. Schwenk, “On the security of modern single sign-on protocols: Second-order vulnerabilities in openid connect,” arXiv Prepr. arXiv1508.04324, 2015.

[19] C. Mainka, V. Mladenov, and J. Schwenk, "Do Not Trust Me: Using Malicious IdPs for Analyzing and Attacking Single Sign-on," 2016 IEEE European Symposium on Security and Privacy (EuroS&P), Saarbruecken, Germany, 2016, pp. 321-336, doi: 10.1109/EuroSP.2016.33.

[20] J. Navas and M. Beltrán, “Understanding and mitigating OpenID Connect threats,” Comput.& Secur., vol. 84, pp. 1–16, 2019, https://doi.org/10.1016/j.cose.2019.03.003.

[21] D. Fett, R. Küsters and G. Schmitz, "The Web SSO Standard OpenID Connect: In-depth Formal Security Analysis and Security Guidelines," 2017 IEEE 30th Computer Security Foundations Symposium (CSF), Santa Barbara, CA, USA, 2017, pp. 189-202, doi: 10.1109/CSF.2017.20.

[22] A. S. K. Maqbool, A. Ali, “Evaluation of OAuth 2.0 Vulnerabilities in Android Applications,” in in Proceedings of the 2020 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM ’20), 2020, pp. 1–6.

[23] A. Kountouras and G. Frantzeskou, “An Empirical Study of Security Misconfiguration in Mobile OAuth.,” in in Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS ’19), 2019, pp. 2283–2300.

[24] J. Richer and A. Sanso, “Security Analysis of OAuth 2.0,” in in Proceedings of the 2015 ACM SIGSAC Conference on Computer and Communications Security (CCS ’15), 2015, pp. 197–208.

[25] B. Braithwaite and A. Doupé, “Breaking the OAuth Protocol,” in in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS ’14), 2014, pp. 676–687.

[26] P. Pereira, T., Felber, P., & Esteves-Veríssimo, “SoK: OAuth 2.0 and Beyond: Current Vulnerabilities, Limitations, and Ongoing Improvements,” in Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P), 2018, pp. 415–430.

[27] K. Küsters, R., & Kifayat, “Formal Security Analysis of OpenID Connect,” in Proceedings of the 2016 IEEE 30th Computer Security Foundations Symposium (CSF), 2016, pp. 189–202.

[28] A. Sanchez-Aguilar, J., Marin, A., & Toval, “Security Modeling of Single Sign-On Protocols: A Comparative Survey,” J. Netw. Comput. Appl., vol. 159, 2020.

[29] S. Vetrivelan, S., Parameswaran, M., & Sen, “Mobile Single Sign-On Using OpenID Connect: A Study on Design and Security Measures,” in Proceedings of the 2019 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), 2019, pp. 1850–1854.

[30] J. Alshehri, S., Nguyen, D., & Indulska, “Empirical Investigation of OpenID Connect: Security, Privacy, and Usability,” in Proceedings of the 2020 IEEE International Conference on Smart Cloud (SmartCloud), 2020, pp. 16–23.

[31] H. Sun, Y., Yang, M., & Gu, “A Survey on OAuth-Based Single Sign-On (SSO) Protocol,” in Proceedings of the 2018 3rd International Conference on Information Science and Systems (ICISS), 2018, pp. 467–472.

[32] M. Z. Khan, A. Y., & Shafiq, “Security Analysis of OAuth 2.0 Framework for Mobile Applications,” in Proceedings of the 2019 10th International Conference on Information and Communication Systems (ICICS), 2019, pp. 48–54.

[33] V. Mladenov and C. Mainka, “OpenID connect–security considerations,” Bochum, January 2017, Ruhr-Univeristät Bochum, 2017.