Man in the Middle Attack in SAML and OpenID Connect

Daniel Muster, www.it-rm.ch, CH 8003 Zürich,
Florian Forster, www.zitadel.com, CH 9000 St. Gallen
6 December 2023, V.1.0

A Man-In-The-Middle (MITM) Attack on OpenID Connect and on a centralised login with SAML will be illustrated in this paper. The attack succeeds if the connection endpoint of the RP (Relying Party) or the SP (Service Provider) does not match with its redirection endpoint and if the IdP or OP does not always show to the user for which RP or Service Provider he will be authenticated.

OpenID Connect and (centralised login with) SAML are standardised IT-protocols for centralised login. Both are widely spread implemented. A method how to attack and how to circumvent these security technologies may have a significant impact, especially if the attack can be successful under circumstances simple to realise and often found in practice.

The principles of the Man in the Middle attack, short MITM-attack, will be illustrated by means of the OpenID Connect Authorization Code Flow protocol [1]. With this explanation it will become clear that the attack will also work with SAML.

The attack profits of two often implemented vulnerabilities (see figure 1):

1. The application and security technology are not cryptographically linked, e.g. TLS Session ID with the http-connection (channel binding), see also “Intent to Remove: Token Binding” [4]
2. IDs for the application and the security technology are different. It is not verified if the IDs correspond with each other. This should be done even if the IDs are used temporarily.

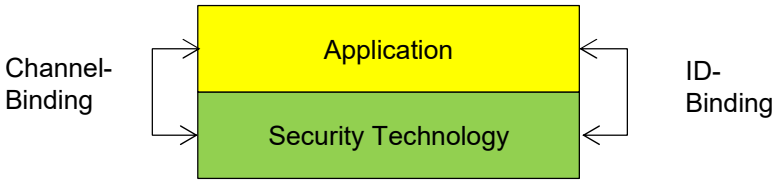

Figure 1

Target audience of this paper: The reader is familiar with OpenID Connect standard/protocol [1], SAML [2] and its application.

Conditions for Success of Attack

Four parties are involved, the OP (OpenID Provider), the RP (Relying Party), the user and the attacker (server).

The success of the MITM attack requires that

1. The connection endpoint of the RP differs from its redirection endpoint. The connections to these endpoints are established each with another TLS session.
2. The user cannot intervene when he will be authenticated and connected to a RP he did not intend to be connected to. E.g. this happens if the IdP does not show to the user who have sent the authentication request. If OpenID Connect or SAML are deployed as centralised login technology then the user often cannot recognise the RP or SP he really will be authenticated for.

Attack Flow on the OpenID Connect Protocol

The procedure of the attack on the OpenID-Connect (Authorization Code Flow) Protocol is illustrated in figure 2. Its explanation follows:

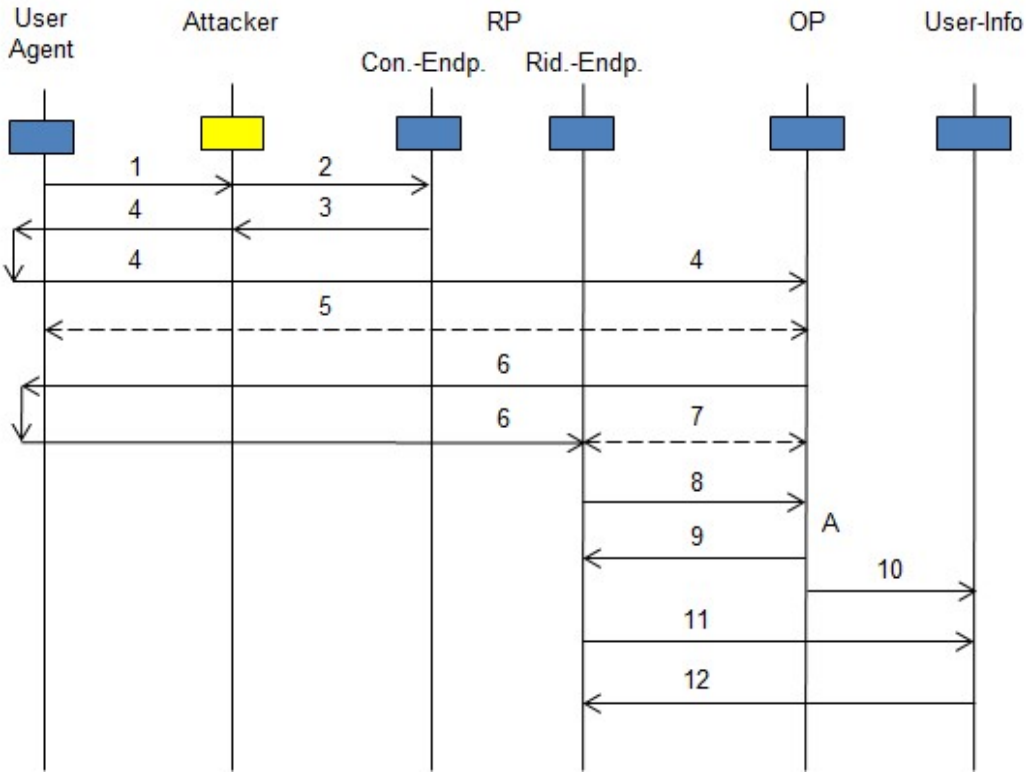

Figure 2. Attack Procedure

Con.-Endp. = Connection Endpoint
Rid.-Endp. = Redirection Endpoint
OP = OpenID Provider
User-Info = User-Info Endpoint

1. The user wants to connect to the server of the attacker, e.g. to an online game server or a shopping server.
2. Immediately after receiving the connection request from the user the attacker sends a connection request to the connection endpoint of the RP. The RP may be a server with sensitive data the user has the authorisation to access.
3. Believing that the connection request has come from a legitimate user the RP sends an *Authentication Request* back to the attacker server.
Authentication Request: response_type=code, client_id, state, redirect_uri, nonce,

code_challenge, code challenge_method. For more information about code_challenge and code_challenge_method see [3].

4. The attacker server transfers the *Authentication Request* unchanged to the User Agent. The User Agent redirects the *Authentication Request* from the RP to the OP.

5. The OP authenticates the user without showing him for whom the user will be authenticated. The user assumes to be connected to the attacker server without knowing that he will be finally authenticated to the RP.

6. The OP sends an *Authentication Response* through the User Agent directly to the redirection endpoint of the RP. This *Authentication Response* does not pass through the attacker server. If the RP accepts the *Authentication Response* then the attack is successful. From now on the OpenID Connect protocol flow doesn't differ from a normal Open ID Connect procedure.
   Authentication Response: redirect_uri, authorization code, state, nonce

7. RP and IdP authenticate each other, best with mTLS (mutual authenticated TLS).

8. RP sends a *Token Request*.
   Token Request: grant_type= authorization_code, client_id, redirect_uri, code = "authorization_code", code_verifier

   A) Verification of the conformity client_id, client_certificate, authorization_code, redirect_uri.

9. OP sends a *Token Response*.
   Token Response: access_token, token_type

10. OP sends to the User-Info endpoint the credentials to authenticate the UserInfo Request sent by the RP.
    user_id, access_token

11. RP sends a *UserInfo Request* to the User-Info endpoint.
    UserInfo Request: access_token

12. User-Info endpoint sends a *UserInfo Response* to the RP. Now the attacker server has access to the RP with equal rights as the user.

Remark: The attack succeeds because the connection of the *Authentication Request* is not (cryptographically) linked with or differs from the connection of the *Authentication response*.

SAML

Even if every message exchanged is signed the attack will be successful under the conditions 1 and 2 mentioned in "Conditions for the Success of the Attack", too. SAML does not have a standardised (and cryptographically binded) link between the redirection (Assertion Consumer Service URL) and the connection endpoint of the Service Provider (name of RP in SAML).

Mitigation/Conclusion

The mitigation consists of the following security measures 1 and 2.

1) Anytime when an *Authentication Request* is sent to the OP or IdP the OP or IdP has to prompt to the user for which RP the user he will be authenticated. At this moment the user may be able to recognise that he will not be authenticated for the attacker server.

Form the value of client_ID the IdP has to extract the corresponding information about the RP in his data base and present it to the user. The information has to be collected at the reg-

istration of the RP and must contain the URL of the RP. Afterwards the user has to verify carefully if this URL matches to URL of server he wants to connect to. If this doesn't match the user must disconnect the session.

That's why the user has to verify carefully the display prompted by the IdP every time he will be authenticated.

2) The session parameters of the RP connection endpoint have to match exactly with those of the RP redirection endpoint, e.g. session ID of the TLS connection, cookie, state. This must be verified by the RP.

Summary

The attack illustrated here is simple to realise but has a significant impact because its premises of success are often fulfilled in practice. Hence it is worth to think about security before deploying OpenID Connect or SAML technologies.

References

[1] OpenID Connect Core 1.0 incorporating errata 1, https://openid.net/specs/openid-connect-core-1_0.html

[2] Security Assertion Markup Language Standards (SAML), https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

[3] RFC 7636 Proof Key Code Exchange by OAuth Public Clients, https://www.rfc-editor.org/search/rfc_search_detail.php?rfc=7636&pubstatus%5B%5D=Any&pub_date_type=any

[4] Intent to Remove Token Binding, https://groups.google.com/a/chromium.org/g/blink-dev/c/OkdLUyYmY1E/m/YJrsadYKDQAJ?pli=1