**ISO #####-#:####(X)**

ISO TC ###/SC ##/WG #

Secretariat: XXXX

**Information technology — OpenID Connect — OAuth 2.0 Form Post Response Mode**

PAS Submission

© ISO 2024

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO’s member body in the country of the requester.

ISO copyright office

CP 401 • Ch. de Blandonnet 8

CH-1214 Vernier, Geneva

Phone: +41 22 749 01 11

Email: copyright@iso.org

Website: www.iso.org

Published in Switzerland

Workgroup:

fapi

Internet-Draft:

fapi-2\_0-attacker-model-03

Published:

7 December 2022

Intended Status:

Standards Track

Author:

D. Fett

*yes.com*

**FAPI 2.0 Attacker Model**

**Abstract**

OIDF FAPI 2.0 is an API security profile suitable for high-security applications based on the OAuth 2.0 Authorization Framework [[RFC6749](#RFC6749)]. This document describes that attacker model that informs the decisions on security mechanisms employed by the FAPI security profiles.[¶](#section-abstract-1)

**Table of Contents**

* [1](#section-1).  [Introduction](#name-introduction)[¶](#section-toc.1-1.1.1)
  + [1.1](#section-1.1).  [Warning](#name-warning)[¶](#section-toc.1-1.1.2.1.1)
  + [1.2](#section-1.2).  [Notational Conventions](#name-notational-conventions)[¶](#section-toc.1-1.1.2.2.1)
* [2](#section-2).  [Security Goals](#name-security-goals)[¶](#section-toc.1-1.2.1)
  + [2.1](#section-2.1).  [Authorization](#name-authorization)[¶](#section-toc.1-1.2.2.1.1)
  + [2.2](#section-2.2).  [Authentication](#name-authentication)[¶](#section-toc.1-1.2.2.2.1)
  + [2.3](#section-2.3).  [Session Integrity](#name-session-integrity)[¶](#section-toc.1-1.2.2.3.1)
* [3](#section-3).  [Attacker Model](#name-attacker-model)[¶](#section-toc.1-1.3.1)
* [4](#section-4).  [Attackers](#name-attackers)[¶](#section-toc.1-1.4.1)
  + [4.1](#section-4.1).  [A1 - Web Attacker](#name-a1-web-attacker)[¶](#section-toc.1-1.4.2.1.1)
  + [4.2](#section-4.2).  [A1a - Web Attacker (participating as AS)](#name-a1a-web-attacker-participat)[¶](#section-toc.1-1.4.2.2.1)
  + [4.3](#section-4.3).  [A2 - Network attacker](#name-a2-network-attacker)[¶](#section-toc.1-1.4.2.3.1)
  + [4.4](#section-4.4).  [Attackers at the Authorization Endpoint](#name-attackers-at-the-authorizat)[¶](#section-toc.1-1.4.2.4.1)
    - [4.4.1](#section-4.4.1).  [A3a - Read Authorization Request](#name-a3a-read-authorization-requ)[¶](#section-toc.1-1.4.2.4.2.1.1)
  + [4.5](#section-4.5).  [Attackers at the Token Endpoint](#name-attackers-at-the-token-endp)[¶](#section-toc.1-1.4.2.5.1)
    - [4.5.1](#section-4.5.1).  [A5 - Read and Tamper with Token Requests and Responses](#name-a5-read-and-tamper-with-tok)[¶](#section-toc.1-1.4.2.5.2.1.1)
  + [4.6](#section-4.6).  [Attackers at the Resource Server](#name-attackers-at-the-resource-s)[¶](#section-toc.1-1.4.2.6.1)
    - [4.6.1](#section-4.6.1).  [A7 - Read Resource Requests](#name-a7-read-resource-requests)[¶](#section-toc.1-1.4.2.6.2.1.1)
* [5](#section-5).  [Limitations](#name-limitations)[¶](#section-toc.1-1.5.1)
  + [5.1](#section-5.1).  [Protocol Layers](#name-protocol-layers)[¶](#section-toc.1-1.5.2.1.1)
  + [5.2](#section-5.2).  [Secrets](#name-secrets)[¶](#section-toc.1-1.5.2.2.1)
  + [5.3](#section-5.3).  [System Boundaries](#name-system-boundaries)[¶](#section-toc.1-1.5.2.3.1)
  + [5.4](#section-5.4).  [Implementation Errors](#name-implementation-errors)[¶](#section-toc.1-1.5.2.4.1)
  + [5.5](#section-5.5).  [Changes over Time](#name-changes-over-time)[¶](#section-toc.1-1.5.2.5.1)
* [6](#section-6).  [Acknowledgements](#name-acknowledgements)[¶](#section-toc.1-1.6.1)
* [7](#section-7).  [Normative References](#name-normative-references)[¶](#section-toc.1-1.7.1)
* [8](#section-8).  [Informative References](#name-informative-references)[¶](#section-toc.1-1.8.1)
* [Appendix A](#section-appendix.a).  [Notices](#name-notices)[¶](#section-toc.1-1.9.1)
* [Author's Address](#name-authors-address)[¶](#section-toc.1-1.10.1)

**1. Introduction**

Since OIDF FAPI 2.0 aims at providing an API protection profile for high-risk scenarios, clearly defined security requirements are indispensable. In this document, the security requirements are expressed through security goals and attacker models. From these requirements, the security mechanisms utilized in the Security Profile are derived.[¶](#section-1-1)

Implementers and users of the Security Profile can derive from this document which threats have been taken into consideration by the Security Profile and which fall outside of what the Security Profile can provide.[¶](#section-1-2)

The ultimate aim is to provide systematic proofs of the security of the FAPI profiles similar to those in [[arXiv.1901.11520](#arXiv.1901.11520)]. Formal proofs can rule out large classes of attacks rooted in the logic of security protocols. Until such proofs are provided for the FAPI 2.0 Security Profile, the attacker model laid out herein informs the design decisions, but, as with most security protocols, there is no guarantee that all attacks for all types of attackers are excluded.[¶](#section-1-3)

The security requirements in this document are expressed in a form that lends itself well to a transfer into a formal representation required for an automated or manual analysis of the security of FAPI. This work draws from the attacker model and security goals formulated in [[arXiv.1901.11520](#arXiv.1901.11520)].[¶](#section-1-4)

**1.1. Warning**

This document is not an OIDF International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.[¶](#section-1.1-1)

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.[¶](#section-1.1-2)

**1.2. Notational Conventions**

The keywords "shall", "shall not", "should", "should not", "may", and "can" in this document are to be interpreted as described in ISO Directive Part 2 [[ISODIR2](#ISODIR2)]. These keywords are not used as dictionary terms such that any occurrence of them shall be interpreted as keywords and are not to be interpreted with their natural language meanings.[¶](#section-1.2-1)

**2. Security Goals**

In the following, the security goals for the FAPI 2.0 Security Profile with regards to authorization and, when OpenID Connect is used, authentication, are defined.[¶](#section-2-1)

[**2.1.**](#section-2.1) [**Authorization**](#name-authorization)

The FAPI 2.0 Security Profile aims to ensure that **no attacker can access protected resources** other than their own.[¶](#section-2.1-1)

The access token is the ultimate credential for access to resources in OAuth. Therefore, this security goal is fulfilled if no attacker can successfully obtain and use an access token for access to protected resources other than their own.[¶](#section-2.1-2)

[**2.2.**](#section-2.2) [**Authentication**](#name-authentication)

The FAPI 2.0 Security Profile aims to ensure that **no attacker is able to log in at a client under the identity of another user.**[¶](#section-2.2-1)

The ID token is the credential for authentication in OpenID Connect. This security goal therefore is fulfilled if no attacker can obtain and use an ID token identifying another user for login.[¶](#section-2.2-2)

[**2.3.**](#section-2.3) [**Session Integrity**](#name-session-integrity)

Session Integrity is concerned with attacks where a user is tricked into logging in under the attacker's identity or inadvertently using the resources of the attacker instead of the user's own resources. Attacks in this field include CSRF attacks (traditionally defended against by using "state" in OAuth) and session swapping attacks.[¶](#section-2.3-1)

In detail:[¶](#section-2.3-2)

* For authentication: The FAPI 2.0 Security Profile aims to ensure that **no attacker is able to force a user to be logged in under the identity of the attacker.**[¶](#section-2.3-3.1.1)
* For authorization: The FAPI 2.0 Security Profile aims to ensure that **no attacker is able to force a user to use resources of the attacker.**[¶](#section-2.3-3.2.1)

[**3.**](#section-3) [**Attacker Model**](#name-attacker-model)

This attacker model defines very broad capabilities for attackers. It is assumed that attackers will exploit these capabilities to come up with attacks on the security goals defined above. To provide a very high level of security, attackers are assumed very powerful, including having access to otherwise encrypted communication.[¶](#section-3-1)

This model does intentionally not define concrete threats. For example, an attacker that has the ability to eavesdrop on an authorization request might be able to use this capability for various types of attacks posing different threats, e.g., injecting a modified authorization request. In a complex protocol like OAuth or OpenID Connect, however, yet unknown types of threats and variants of existing threats can emerge, as has been shown in the past. In order to not overlook any potential attacks, FAPI 2.0 therefore aims not to address concrete, narrow threats, but to exclude any attacks conceivable for the attacker types listed here. This will be supported by a formal analysis, as mentioned above.[¶](#section-3-2)

This attacker model assumes that certain parts of the infrastructure are working correctly. Failures in these parts likely lead to attacks that are out of the scope of this attacker model. These areas need to be analyzed separately within the scope of an application of the FAPI 2.0 security profiles using threat modelling or other techniques.[¶](#section-3-3)

For example, if a major flaw in TLS was found that undermines data integrity in TLS connections, a network attacker (A2, below) would be able to compromise practically all OAuth and OpenID Connect sessions in various ways. This would be fatal, as even application-level signing and encryption is based on key distribution via TLS connections. As another example, if a human error leads to the disclosure of secret keys for authentication and an attacker would be able to misuse these credentials, this attack would not be covered by this attacker model.[¶](#section-3-4)

The following parts of the infrastructure are out of the scope of this attacker model:[¶](#section-3-5)

* **TLS:** It is assumed that TLS connections are not broken, i.e., data integrity and confidentiality are ensured. The correct public keys are used to establish connections and private keys are not known to attackers (except for explicitly compromised parties).[¶](#section-3-6.1.1)
* **JWKS:** Where applicable, key distribution mechanisms work as intended, i.e., encryption and signature verification keys of uncompromised parties are retrieved from the correct endpoints.[¶](#section-3-6.2.1)
* **Browsers and Endpoints:** Devices and browsers used by resource owners are considered not compromised. Other endpoints not controlled by an attacker behave according to the protocol.[¶](#section-3-6.3.1)

[**4.**](#section-4) [**Attackers**](#name-attackers)

FAPI 2.0 profiles aim to ensure the security goals listed above for arbitrary combinations of the following attackers, potentially collaborating to reach a common goal:[¶](#section-4-1)

[**4.1.**](#section-4.1) [**A1 - Web Attacker**](#name-a1-web-attacker)

Standard web attacker model. Can send and receive messages just like any other party controlling one or more endpoints on the internet. Can participate in protocols flows as a normal user. Can use arbitrary tools (e.g., browser developer tools, custom software, local interception proxies) on their own endpoints to tamper with messages and assemble new messages. Can send links to honest users that are then visited by these users. This means that the web attacker has the ability to cause arbitrary requests from users' browsers, as long as the contents are known to the attacker.[¶](#section-4.1-1)

Cannot intercept or block messages sent between other parties, and cannot break cryptography unless the attacker has learned the respective decryption keys. Deviating from the common web attacker model, A1 cannot play the role of a legitimate AS in the ecosystem (see A1a).[¶](#section-4.1-2)

[**4.2.**](#section-4.2) [**A1a - Web Attacker (participating as AS)**](#name-a1a-web-attacker-participat)

Like the web attacker A1, but can also participate as an AS in the ecosystem. Note that this AS can reuse/replay messages it has received from honest ASs and can send users to endpoints of honest ASs.[¶](#section-4.2-1)

[**4.3.**](#section-4.3) [**A2 - Network attacker**](#name-a2-network-attacker)

Controls the whole network (like a rogue WiFi access point or any other compromised network node). Can intercept, block, and tamper with messages intended for other people, but cannot break cryptography unless the attacker has learned the respective decryption keys.[¶](#section-4.3-1)

Note: Most attacks that are exclusive to this kind of attacker can be defended against by using transport layer protection like TLS.[¶](#section-4.3-2)

[**4.4.**](#section-4.4) [**Attackers at the Authorization Endpoint**](#name-attackers-at-the-authorizat)

**Note:** The attackers for the authorization request are more fine-grained than those for the token endpoint and resource endpoint, since these messages pass through the complex environment of the user's browser/app/OS with a larger attack surface. This demands for a more fine-grained analysis.[¶](#section-4.4-1)

**Note:** For the authorization endpoint, it is assumed that the attacker can only passively read messages, whereas for the token and resource endpoints, it is assumed that the attacker can also tamper with messages. Since messages to and from the authorization endpoint are sent through the user's browser and the attacker can redirect the user to arbitrary URLs anyway (see A1), the attacker can already redirect the user to faked/spoofed authorization request and response URLs. At the same time, while leakages from the authorization request or response are very common in practice, a fully compromised connection to the authorization endpoint is not. Most user authentication schemes would be broken in this setting, undermining the security completely.[¶](#section-4.4-2)

[**4.4.1.**](#section-4.4.1) [**A3a - Read Authorization Request**](#name-a3a-read-authorization-requ)

The capabilities of the web attacker, but can also read the authorization request sent in the front channel from a user's browser to the authorization server. This might happen on mobile operating systems (where apps can register for URLs), on all operating systems through the browser history, or due to Cross-Site Scripting on the AS. There have been cases where anti-virus software intercepts TLS connections and stores/analyzes URLs.[¶](#section-4.4.1-1)

Note: An attacker that can read the authorization response is not considered here, as, with current browser technology, such an attacker can undermine most security protocols. This is discussed in "Browser Swapping Attacks" in the Security Considerations in the FAPI 2.0 Security Profile.[¶](#section-4.4.1-2)

[**4.5.**](#section-4.5) [**Attackers at the Token Endpoint**](#name-attackers-at-the-token-endp)

[**4.5.1.**](#section-4.5.1) [**A5 - Read and Tamper with Token Requests and Responses**](#name-a5-read-and-tamper-with-tok)

This attacker makes the client use a token endpoint that is not the one of the honest AS. This attacker can read and tamper with messages sent to and from this token endpoint that the client thinks as of an honest AS.[¶](#section-4.5.1-1)

Important: This attacker is a model for misconfigured token endpoint URLs that were considered in FAPI 1.0. Since the FAPI 2.0 Security Profile mandates that the token endpoint address is obtained from an authoritative source and via a protected channel, i.e., through OAuth Metadata obtained from the honest AS, this attacker is not relevant in FAPI 2.0. The description here is kept for informative purposes only.[¶](#section-4.5.1-2)

[**4.6.**](#section-4.6) [**Attackers at the Resource Server**](#name-attackers-at-the-resource-s)

[**4.6.1.**](#section-4.6.1) [**A7 - Read Resource Requests**](#name-a7-read-resource-requests)

The capabilities of the web attacker, but this attacker can also read requests sent to the resource server after they have been processed by the resource server, for example because the attacker can read TLS intercepting proxy logs on the RS's side.[¶](#section-4.6.1-1)

[**5.**](#section-5) [**Limitations**](#name-limitations)

Beyond the limitations already described in the introduction to the attacker model above, it is important to note the following limitations:[¶](#section-5-1)

[**5.1.**](#section-5.1) [**Protocol Layers**](#name-protocol-layers)

FAPI 2.0 profiles only define the behavior of API authorization and authentication on certain protocol layers. As described above, attacks on lower protocol layers (e.g., TLS) may break the security of FAPI 2.0 compliant systems under certain conditions. The attacker model, however, takes some breaks in the end-to-end security provided by TLS into account by already including the respective attacker models (A3a/A5/A7). Similarly, many other attacks on lower layers are already accounted for, for example:[¶](#section-5.1-1)

* DNS spoofing attacks are covered by the network attacker (A2)[¶](#section-5.1-2.1.1)
* Leakages of authorization request data, e.g., through misconfigured URLs or system/firewall logs, are covered by A3a[¶](#section-5.1-2.2.1)
* Directing users to malicious websites is within the capabilities of the web attacker (A1)[¶](#section-5.1-2.3.1)

FAPI 2.0 aims to be secure when attackers exploit these attacks and all attacks feasible to attackers described above, even in combination.[¶](#section-5.1-3)

Other attacks are not covered by the attacker model. For example, user credentials being exposed through misconfigured databases or remote code execution attacks on authorization servers are neither prevented by nor accounted for in the attacker model. As another example, when a user is using a compromised browser and operating system, the security of the user is hard to uphold. Phishing-resistant credentials, for example, may help in this case, but are outside of the area defined by FAPI 2.0, as described next.[¶](#section-5.1-4)

[**5.2.**](#section-5.2) [**Secrets**](#name-secrets)

The security assessment assumes that secrets are created such that attackers cannot guess them - e.g., nonces and secret keys. Weak random number generators, for example, may lead to secrets that are guessable by attackers and therefore to vulnerabilities.[¶](#section-5.2-1)

[**5.3.**](#section-5.3) [**System Boundaries**](#name-system-boundaries)

The FAPI 2.0 profiles focus on core aspects of the API security and do not prescribe, for example, end-user authentication mechanisms, firewall setups, software development practices, or security aspects of internal architectures. Anything outside of boundaries of FAPI 2.0 must be assessed in the context of the ecosystem, deployment, or implementation in which FAPI 2.0 is used.[¶](#section-5.3-1)

[**5.4.**](#section-5.4) [**Implementation Errors**](#name-implementation-errors)

API security profiles can define how authentication and authorization is supposed to be implemented and a formal model can assess whether the profiles are secure and consistent with respect to ideal implementations. Real-world implementations, of course, can deviate from the specified and formally analyzed behavior and contain security vulnerabilties on various levels. While the FAPI 2.0 profiles are designed to provide multiple layers of defense where feasible, implementations must use secure software development and deployment best practices to ensure that vulnerabilities can be discovered and fixed.[¶](#section-5.4-1)

[**5.5.**](#section-5.5) [**Changes over Time**](#name-changes-over-time)

New technologies or changed behavior of components (e.g., browsers) can lead to new security vulnerabilities over time that might not have been known during the development of these specifications.[¶](#section-5.5-1)

[**6.**](#section-6) [**Acknowledgements**](#name-acknowledgements)

We would like to thank Dave Tonge, Nat Sakimura, Brian Campbell, Torsten Lodderstedt, Joseph Heenan, Pedram Hosseyni, Ralf Küsters and Tim Würtele for their valuable feedback and contributions that helped to evolve this document.[¶](#section-6-1)

[**7.**](#section-7) [**Normative References**](#name-normative-references)

**[RFC6749]**

Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <<https://www.rfc-editor.org/info/rfc6749>>.

**[ISODIR2]**

Standardization, I. O. F., "ISO/IEC Directives Part 2 -", , <<https://www.iso.org/sites/directives/current/part2/index.xhtml>>.

[**8.**](#section-8) [**Informative References**](#name-informative-references)

**[arXiv.1901.11520]**

Fett, D., Hosseyni, P., and R. Küsters, "An Extensive Formal Security Analysis of the OpenID Financial-grade API", 31 January 2019, <<http://arxiv.org/abs/1901.11520/>>.

[**Appendix A.**](#section-appendix.a) [**Notices**](#name-notices)

Copyright (c) 2022 The OpenID Foundation.[¶](#section-appendix.a-1)

The OpenID Foundation (OIDF) grants to any Contributor, developer, implementer, or other interested party a non-exclusive, royalty free, worldwide copyright license to reproduce, prepare derivative works from, distribute, perform and display, this Implementers Draft or Final Specification solely for the purposes of (i) developing specifications, and (ii) implementing Implementers Drafts and Final Specifications based on such documents, provided that attribution be made to the OIDF as the source of the material, but that such attribution does not indicate an endorsement by the OIDF.[¶](#section-appendix.a-2)

The technology described in this specification was made available from contributions from various sources, including members of the OpenID Foundation and others. Although the OpenID Foundation has taken steps to help ensure that the technology is available for distribution, it takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this specification or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any independent effort to identify any such rights. The OpenID Foundation and the contributors to this specification make no (and hereby expressly disclaim any) warranties (express, implied, or otherwise), including implied warranties of merchantability, non-infringement, fitness for a particular purpose, or title, related to this specification, and the entire risk as to implementing this specification is assumed by the implementer. The OpenID Intellectual Property Rights policy requires contributors to offer a patent promise not to assert certain patent claims against other contributors and against implementers. The OpenID Foundation invites any interested party to bring to its attention any copyrights, patents, patent applications, or other proprietary rights that may cover technology that may be required to practice this specification.[¶](#section-appendix.a-3)

[**Author's Address**](#name-authors-address)

Daniel Fett

yes.com

Email: [mail@danielfett.de](mailto:mail@danielfett.de)