|  |  |  |  |
| --- | --- | --- | --- |
| Document filename: | **NHS login Interface Specification - Federation** | | |
| Directorate / Programme | **NHS login Programme** |  |  |
| Document Reference | **CID-IS-F** | Status | **DRAFT** |
| Information Asset Owner | Melissa Ruscoe | Version | **V1.10** |
| Author | Matthew Brown | Version issue date | 29 Sept 2020 |

**NHS login**

**Interface Specification - Federation**

**Document Management**

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Summary of Changes** |
| V1.0 Draft | 30 April 2018 | First draft for review |
| V1.1 Draft | 16 July 2018 | Second draft |
| V1.2 Draft | 30 July 2018 | Draft following comments |
| V1.3 Draft | 23 Aug 2018 | Draft for informal external comments |
| V1.4 Draft | 06 Nov 2018 | Draft for external supplier comments |
| V1.4i | 19 Mar 2019 | Updated following comments. Issued for approval for Public Beta |
| V1.5 | 01 Apr 2019 | Include refresh token |
| V1.6 | 07 May 2019 | Include FIDO UAF details |
| V1.7 | 21 Oct 2019 | Minor clarifications on supported scope and claims |
| V1.8 | 19 Nov 2019 | Include client\_metadata scope |
| V1.9 | 15 May 2020 | Include identity\_proofing\_level claim and change to section 5.1.1, 5.1.3 & 7.3 |
| V1.10 | 29 Sept 20 | Minor changes related to vot |

**Reviewers**

This document must be reviewed by the following people:

|  |  |  |  |
| --- | --- | --- | --- |
| **Reviewer name** | **Title / Responsibility** | **Date** | **Version** |
| Matthew Brown | Lead Technical Architect | 29 Sept 2020 | V1.10 |
| Priyanka Mittal | NHS login Technical Architect | 29 Sept 2020 | V1.10 |

**Approved by**

This document must be approved by the following people:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Title / Responsibility** | **Date** | **Version** |
| Melissa Ruscoe | Programme Head | 29 Sept 2020 | V1.10 |
| Darren Hutton | Product Owner | 29 Sept 2020 | V1.10 |

**Document Control:**

The controlled copy of this document is maintained in the NHS Digital corporate network. Any copies of this document held outside of that area, in whatever format (e.g. paper, email attachment), are considered to have passed out of control and should be checked for currency and validity.

Contents

[1 Introduction 6](#_Toc40345294)

[1.1 Purpose of Document 6](#_Toc40345295)

[1.2 Audience 6](#_Toc40345296)

[1.3 Definitions 6](#_Toc40345297)

[2 Specification Status 7](#_Toc40345298)

[2.1 Scope / Constraints 7](#_Toc40345299)

[2.2 Changes in this version 7](#_Toc40345300)

[3 Messages Overview 8](#_Toc40345301)

[3.1 Authorization Code Flow 8](#_Toc40345302)

[3.2 Public and Confidential Clients 10](#_Toc40345303)

[3.3 Endpoints 11](#_Toc40345304)

[3.4 Authorize Endpoint 12](#_Toc40345305)

[3.5 Token Endpoint 21](#_Toc40345306)

[3.6 UserInfo Endpoint 25](#_Toc40345307)

[4 Tokens 32](#_Toc40345308)

[4.1 JWT Header 32](#_Toc40345309)

[4.2 ID Token Payload 32](#_Toc40345310)

[4.3 Access Token Payload 36](#_Toc40345311)

[4.4 JOSE Signing 38](#_Toc40345312)

[5 Data View 39](#_Toc40345313)

[5.1 Vectors of Trust 39](#_Toc40345314)

[6 Password-less Authentication using FIDO UAF 45](#_Toc40345315)

[6.1 Overview 45](#_Toc40345316)

[6.2 FIDO UAF registration flow 46](#_Toc40345317)

[6.3 FIDO UAF authentication flow 50](#_Toc40345318)

[6.4 FIDO UAF deregistration flow 52](#_Toc40345319)

[6.5 Platform Restrictions 54](#_Toc40345320)

[7 Partner Services and Security 56](#_Toc40345321)

[7.1 Partner Service Registration 56](#_Toc40345322)

[7.2 Partner Service Authentication 56](#_Toc40345323)

[7.3 Client Redirects 57](#_Toc40345324)

[8 References 58](#_Toc40345325)

# Introduction

## Purpose of Document

NHS login implements OpenID Connect 1.0 OpenID Provider role to assert the identity of the End-User to a Partner Service, as well as enabling the Service to obtain basic profile information about the End-User in an interoperable manner.

This document defines the interfaces implemented by the Platform and describes the data flows supported by these interfaces.

## Audience

The primary audiences for this document are:

* NHS login Programme team
* NHS Digital – Digital Delivery Centre
* NHS Digital – Other Delivery teams
* NHS England
* 3rd Party Suppliers integrating with the NHS Digital NHS login Platform

## Definitions

Where used in this document set, the keywords MUST, SHOULD and MAY are to be interpreted as follows:

* **MUST**: This word, or the terms "**REQUIRED**" or "**SHALL**", means that the definition is an absolute` requirement of the specification.
* **SHOULD**: This word, or the adjective "**RECOMMENDED**", means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications **MUST** be understood and carefully weighed before choosing a different course.
* **MAY**: This word, or the adjective “**OPTIONAL**”, means that an item is truly optional. One implementer may choose to include the item because a particular implementation requires it or because the implementer feels that it enhances the implementation while another implementer may omit the same item. An implementation which does not include a particular option **MUST** be prepared to interoperate with another implementation which does include the option, though perhaps with reduced functionality. In the same vein an implementation which does include a particular option **MUST** be prepared to interoperate with another implementation which does not include the option (except, of course, for the feature the option provides).

# Specification Status

## Scope / Constraints

* This document relates to NHS login ‘Federation interface’ during Beta. Further amendments WILL be implemented prior to full rollout – this document is a draft representing intended design, but is subject to clarification.
* The NHS login Platform supports Authentication using the Authorization Code Flow only
  + Implicit & Hybrid Flows are not supported
* HTTP is not supported - all HTTP-based flows must be HTTPS (using TLS v1.2 or above)

## Changes in this version

Version 1.9 of this specification includes the following changes:

* Addition of identity\_proofing\_level claim in the “profile” scope
* Minor clarification on conditions applicable for requesting scopes
* Updates to Verification of Identity Levels
* Updates to Requesting Vectors of Trust values
* Updates to Client Redirects
* Minor reference changes

# Messages Overview

References:

* OpenID Connect Core Specification [1], section 3.1
* International Government Assurance Profile (iGov) [2]
* Vectors of Trust [3]

This section describes how the NHS Digital NHS login Platform implements authentication using the OpenID Connect Authorization Code Flow.

## Authorization Code Flow

The Authorization Code Flow returns an Authorization Code to the Client, which can then exchange it for an ID Token and an Access Token directly. This avoids exposing any tokens to the User Agent and possibly other malicious applications with access to the User Agent.

Using this flow also means the Platform can also authenticate the Client before exchanging the Authorization Code for an Access Token. The Authorization Code flow is suitable for Clients that can securely maintain a Client Secret between themselves and the Platform

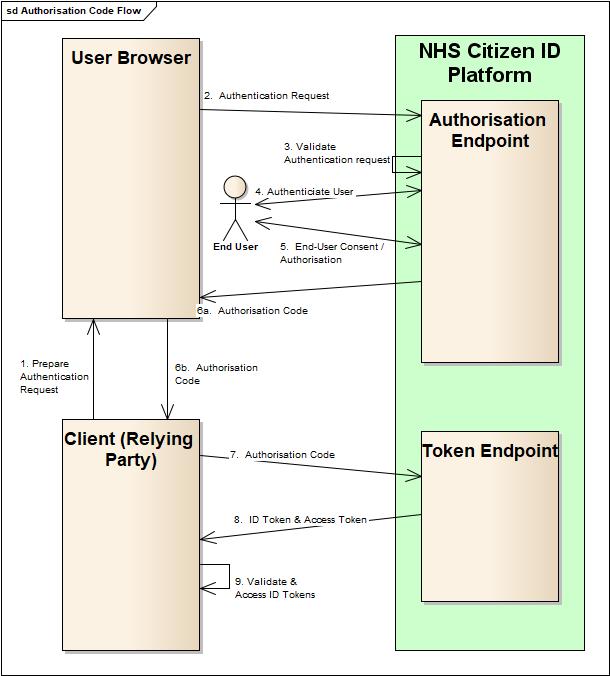


Figure 1: Authorisation Code Flow

The Authorization Code Flow goes through the following steps:

1. The client prepares an Authentication Request containing the desired request parameters (see section 3.4.1
2. The client sends the request to the Platform – via the User Agent (browser)
3. The Platform validates the Authentication Request (see section 3.4.2)
4. The Platform authenticates the End-User (see section 3.4.3)
5. The Platform obtains End-User Consent/Authorization (see section 3.4.4).
6. The Platform sends the Authorization Code to the Client via the End User’s browser
7. The client requests a response using the Authorization Code at the Token Endpoint (see section 3.5.1).
8. The client receives a response that contains an ID Token and Access Token in the response body (see section 3.5.3).
9. The client validates the ID token and retrieves the End-User’s Subject Identifier.

## Public and Confidential Clients

Clients are classified as either Public or Confidential based on their ability to maintain the confidentiality of their credentials.

### Confidential Client

The client is capable of maintaining the confidentiality of its credentials. For example, the client is implemented on a secure server with restricted access to its credentials

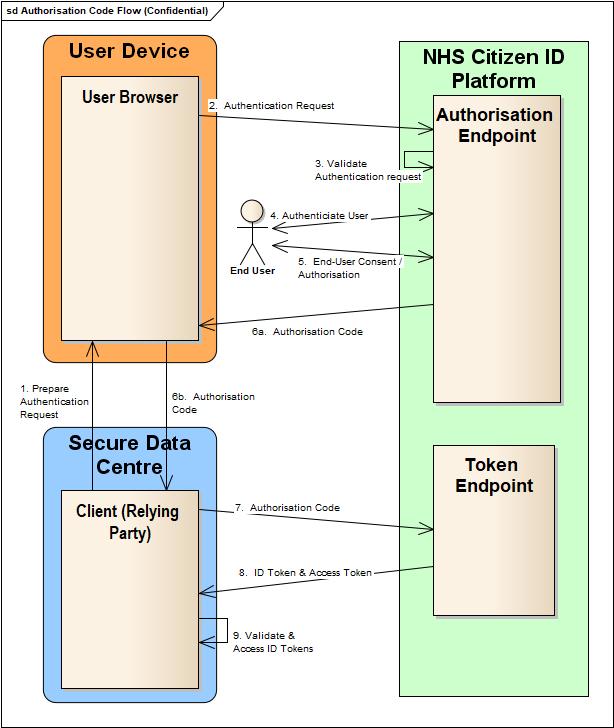


Figure 2: Confidential Client

### Public Client

**This section is provided for information only; public clients are not supported by NHS login**.

A Public client is incapable of maintaining the confidentiality of its credentials. For example, the client executes on a device used by the end-user, such as an installed native application or browser-based application.

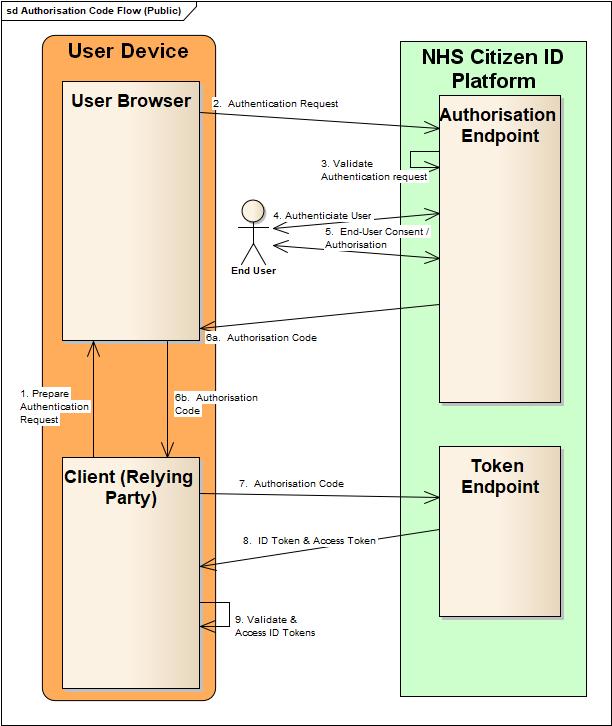


Figure 3: Public Client

Public clients using the Authorization Code flow are susceptible to authorization code interception attacks. For example, a rogue application could intercept the authorization code as it is being passed through the mobile/native operating system. To mitigate this attack, public clients follow the Proof Key for Code Exchange (PKCE) specification, which binds the Authorisation Request to the subsequent Token Request. PKCE is an extension to the regular Authorization Code flow, so the flow is the same, except that PKCE elements are included at various steps.

Public clients are also at risk of masquerading by malicious services; a malicious service could re-use the client\_id of a genuine Public client and trick a user into revealing their information.

**For the above reasons Public clients are not supported by NHS login**.

## Endpoints

/.well-known/openid-configuration – This is the standard openid discovery address – this will maintain full addresses for the below endpoints. This is consumed by a Service with a direct HTTPS call.

/.well-known/jwks.json – The Keys used by NHS login to sign tokens are available here. This is consumed by a secure Service’s backend with a direct HTTPS call.

/authorize – This initiates the Authorization Code flow – the endpoint is consumed by directing a user-agent (typically a web browser) to the endpoint address, using HTTPS. This is also termed the OIDC Authentication Endpoint.

/token – This is used to retrieve tokens, including ID tokens, by exchanging an authorization code. This is consumed by a secure Service’s backend with a direct HTTPS call. The token endpoint is also used to refresh access tokens, using a previously-issued refresh token

/userinfo – This endpoint is used after retrieving tokens – sensitive and/or time-bound attributes are available from this endpoint. This is consumed by a secure Service’s backend with a direct HTTPS call.

## Authorize Endpoint

### Authentication Request

References:

* OpenID Connect Core Specification [1], section 3.1.2.1
* International Government Assurance Profile (iGov) for OpenID Connect [2]
* RFC7519: JSON Web Token (JWT) [4]

The client initiates an authentication request to the NHS Digital NHS login authorize endpoint using the HTTP GET or POST methods. If using HTTP GET, then the parameters are serialised using URI query Serialisation. In the case of HTTP POST, then the parameters are serialised using Form Serialisation (HTTP Content-Type application/x-www-form-urlencoded)

Authentication Request parameters are:

Table 1: Authentication Request Parameters

| Parameter | Req? | Value(s) | Description |
| --- | --- | --- | --- |
| scope | Mand | “openid” | The openid scope value MUST be present.  Other scope values listed MAY be present. Scope values used that are not understood by NHS login WILL be ignored.  Additional scope values supported by NHS login are defined in section 3.4.1.1 |
| response\_type | Mand | “code” | OAuth 2.0 Response Type value that determines the authorization processing flow to be used, including what parameters are returned from the endpoints used. When using the Authorization Code Flow, this value MUST be “code” |
| client\_id | Mand |  | Oauth 2.0 Client Identifier  This is a static identifier previously provided by the NHS login Partner Onboarding team |
| redirect\_uri | Mand |  | Redirection URI to which the response will be sent. This URI MUST exactly match one of the Redirection URI values for the Client pre-registered at the OpenID Provider  When using this flow, the Redirection URI MUST NOT use the http scheme |
| state | Mand |  | Opaque value used to maintain state between the request and the callback. Typically, Cross-Site Request Forgery (CSRF, XSRF) mitigation is done by cryptographically binding the value of this parameter with a browser cookie  This value will be returned to the client in the authentication response.  The iGov profile for OIDC specifies this parameter as Mandatory to help RPs protect against CSRF attacks. |
| response\_mode | n/a |  | The only response\_mode supported by NHS login will be the default mode (“query”), where the Authorization Response parameters are encoded in the query string which is added to the redirect\_uri when redirecting back to the Client  See Oauth 2.0 Multiple Response Type Encoding Practices [5] |
| nonce | Mand |  | String value used to associate a Client session with an ID Token, and to mitigate replay attacks. The value is passed through unmodified from the Authentication Request to the ID Token. Sufficient entropy MUST be present in the nonce values used to prevent attackers from guessing values  The iGov profile for OIDC specifies this parameter as Mandatory to help RPs protect against CSRF attacks. |
| display | Opt | “page”  “touch” | ASCII string value that specifies how the Platform displays the authentication and consent user interface pages to the End-User. The defined values shown “page” and “touch” are supported for NHS login beta phases, with the default value being “page”.  “popup” and “wap” values are not supported |
| prompt | Opt | <blank>  “none”  “login” | Requests that the NHS login Service forces the user to sign-in, or to request that the Service does not prompt the user to sign-in (SSO)  <blank> - The Service will SSO the user if they still have a valid session, else the user will be requested to login  none – The Service will SSO the user if they still have a valid session, otherwise an error code is returned  login – The Service will request the user to login, regardless of a session already existing |
| max\_age | n/a |  | Not currently supported |
| ui\_locales | n/a |  | Not currently supported |
| id\_token\_hint | n/a |  | Not currently supported |
| login\_hint | n/a |  | Not currently supported |
| acr\_values | n/a |  | Not supported – not used, superseded by vtr |
| vtr | Opt | See s5.1 for values | Vector of Trust Request – requested levels of Identity Verification and Authentication.  Client SHOULD request only single level of Identity Verification (values defined in section [5.1.1 -Verification of Identity Levels](#_Verification_of_Identity) and Table 18)  Omission of vtr will result in a default value of “[“P9.Cp.Cd”,“P9.Cp.Ck”,“P9.Cm”]” being assumed |
| **NHS login extensions** | | | |
| fido\_auth\_response | Opt |  | Base64 URL-encoded FIDO UAF AuthResponse message generated by FIDO client on a registered device |
| asserted\_login\_identity | Opt |  | The purpose of this parameter is to support seamless login between two RPs (RP1 and RP2) where cookie-based SSO is not available. The content will be a signed jwt with payload containing “code” attribute with the value being that of the “jti” attribute from the ID Token issued to RP1. The jwt “iss” attribute MUST contain the client\_id of RP1, the jwt MUST have an “exp” of no longer that 60 seconds, MUST have “jti” and “iat” attributes (as per RFC7519) and MUST be signed by RP1 using its client private key. RP1 passes the jwt to RP2 for RP2 to use in its authentication request.  A non-normative example jwt payload section is as follows  {  code: “eeroifoteiwrudjdwusdu”,  iss: “client1”,  jti: “reioteotijdvorijevoihroi”,  iat: 1548701645,  exp: 1548701705  } |
| allow\_registration | Opt | “false” | Optional parameter which, if set to “false”, will hide links to account registration screens in the NHS login UI. If the parameter is absent or set to any value other than “false” then account registration options will be displayed to the user in the NHS login UI. |

#### Scope to be requested

References:

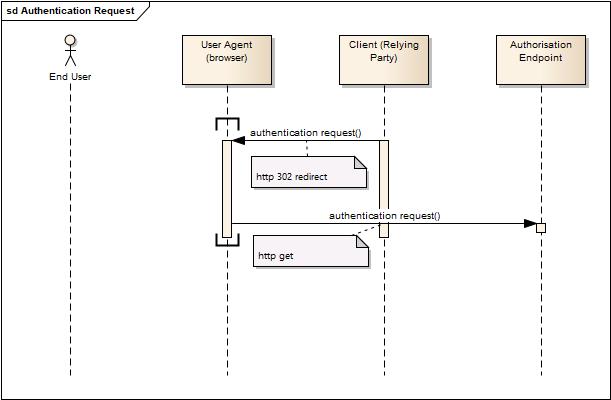
* OpenID Connect Core Specification [1], section 5.1

Scopes can be used to request that specific sets of information be made available as Claim Values when making an Authentication Request (s3.4.1).

Table 2: Scope Values

| Scope | Description |
| --- | --- |
| **OpenID Connect Standard Scopes** | |
| openid | Mandatory value for OpenID Connect Requests |
| profile | This scope value requests access to the End-User’s default profile claims, which are: nhs\_number, birthdate, family\_name, identity\_proofing\_level |
| email | This scope value requests access to the email and email\_verified claims |
| phone | This scope value requests access to the phone\_number and phone\_number\_verified claims |
| **NHS login extensions** | |
| gp\_integration\_credentials | This scope value requests access to the End-User’s gp\_integration\_credentials claims.  Note: This scope can only be requested by an IM1 enabled client and protected by 2FA (two factor authentication) |
| gp\_registration\_details | This scope value requests access to the End-User’s gp\_registration\_details claims as held within the NHS Personal Demographics Service |
| profile\_extended | This scope value requests access to the End-User’s additional demographics claims (as held within the NHS Personal Demographics Service), which are: given\_name |
| client\_metadata | This scope requests access to the End-User’s client\_user\_metadata claim which holds client-specific metadata for the user account. Unlike other claims, the client\_user\_metadata can be updated by the client – see section 3.6.3 and section [3.6.2](#_UserInfo_Response) for more info. |

**Non-Normative Example**

****

An HTTP 302 redirect response by the Client triggers the User Agent (browser) to make an Authentication Request to the Authorization Endpoint (with line wraps within values for display purposes only):

HTTP/1.1 302 Found

Location: https://auth.login.nhs.uk/authorize?

response\_type=code

&scope=openid%20profile%20email

&client\_id=s6BhdRkqt3

&state=af0ifjsldkj

&redirect\_uri=https%3A%2F%2Fclient.example.org%2Fcb

&vtr=%5B%22P9.Cp.Cd%E2%80%9D%5D

The User Agent sends the following request to the Platform in response to the HTTP 302 redirect response by the Client (with line wraps within values for display purposes only):

GET /authorize?

response\_type=code

&scope=openid%20profile%20email

&client\_id=s6BhdRkqt3

&state=af0ifjsldkj

&redirect\_uri=https%3A%2F%2Fclient.example.org%2Fcb

&vtr=%5B%22P9.Cp.Cd%E2%80%9D%5D

HTTP/1.1

Host: https://auth.login.nhs.uk

### Authentication Request Validation

References:

* OpenID Connect Core Specification [1], section 3.1.2.2

The Authentication Request is authenticated as described in OpenID Connect Core Specification [1], s3.1.2.2

Additional validations are:

* the submitted redirect URI is an exact match of the one of the redirection URI values registered for the client

### Authorisation Server Authenticates the End-User

Outside the scope of this spec

### Authorisation Server obtains End-User Consent / Authorisation

Outside the scope of this spec

### Authentication Response

#### Successful response

References:

* OpenID Connect Core Specification [1], s3.1.2.5
* RFC6749 - The OAuth 2.0 Authorization Framework [6], s4.1.2

The Authentication Response returns the parameters listed in Table 3 by adding them as query parameters to the redirect\_uri specified in the Authentication Request.

Table 3: Authentication Successful Response

| Parameter | Req? | Description |
| --- | --- | --- |
| code | Mand | The authorization code generated by the NHS login Platform.  The authorization code SHALL expire shortly after it is issued to mitigate the risk of leaks. A maximum authorization code lifetime of 10 minutes is RECOMMENDED.  The client MUST NOT use the authorization code more than once. If an authorization code is used more than once, the NHS login Platform WILL deny the request.  The authorization code is bound to the client identifier and redirection URI. |
| state | Mand | if the "state" parameter was present in the client authorization request, then it SHALL be the exact value received from the client, otherwise not included |

A non-normative example follows:

HTTP/1.1 302 Found

Location: https://client.example.org/cb?

code=SplxlOBeZQQYbYS6WxSbIA

&state=af0ifjsldkj

#### Error Response

References:

* OpenID Connect Core Specification [1], s3.1.2.6
* RFC6749 - The OAuth 2.0 Authorization Framework [6], s4.1.2.1

If the authentication request is denied or fails, the authorisation server informs the client using the parameters defined in Table 4. Unless the Redirection URI is invalid, the Authorisation Server returns the client to the Redirection URI specified in the Authentication Request with the error and state parameters. Other parameters should not be returned

Table 4: Authentication Error Response

| Parameter | Req? | Description |
| --- | --- | --- |
| error | Mand | Error code, see Table 5 for permitted values |
| error\_description | Opt | Human-readable ASCII encoded text description of the error |
| error\_uri | Opt | URI of a web page that includes additional information about the error |
| state | Cond | OAuth 2.0 state value. REQUIRED if the Authorization Request included the state parameter. Set to the value received from the Client. |

Table 5: Error Codes for Authentication Error Response

| Code | Description |
| --- | --- |
| **OAuth 2.0 error codes** | |
| invalid\_request | The request is missing a required parameter, includes an invalid parameter value, includes a parameter more than once, or is otherwise malformed. |
| unauthorized\_client | The client is not authorized to request an authorization code using this method |
| access\_denied | The resource owner or Platform denied the request |
| unsupported\_response\_type | The Platform does not support obtaining an authorization code using this method |
| invalid\_scope | The requested scope is invalid, unknown, or malformed |
| server\_error | The Platform encountered an unexpected condition that prevented it from fulfilling the request. (This error code is needed because a 500 Internal Server Error HTTP status code cannot be returned to the client via an HTTP redirect.) |
| temporarily\_unavailable | The Platform is currently unable to handle the request due to a temporary overloading or maintenance of the server. (This error code is needed because a 503 Service Unavailable HTTP status code cannot be returned to the client via an HTTP redirect.) |
| **OpenID Connect Error Codes (in addition to the above)** | |
| interaction\_required | The Platform requires End-User interaction of some form to proceed. This error MAY be returned when the prompt parameter value in the Authentication Request is none, but the Authentication Request cannot be completed without displaying a user interface for End-User interaction. |
| login\_required | The Platform requires End-User authentication. This error MAY be returned when the prompt parameter value in the Authentication Request is none, but the Authentication Request cannot be completed without displaying a user interface for End-User authentication. |
| account\_selection\_required | The End-User is REQUIRED to select a session at the Platform. The End-User MAY be authenticated at the Platform with different associated accounts, but the End-User did not select a session. This error MAY be returned when the prompt parameter value in the Authentication Request is none, but the Authentication Request cannot be completed without displaying a user interface to prompt for a session to use |
| consent\_required | The Platform requires End-User consent. This error MAY be returned when the prompt parameter value in the Authentication Request is none, but the Authentication Request cannot be completed without displaying a user interface for End-User consent |
| invalid\_request\_uri | The request\_uri in the Authorization Request returns an error or contains invalid data |
| invalid\_request\_object | The request parameter contains an invalid Request Object |
| request\_not\_supported | The OP does not support use of the request parameter |
| request\_uri\_not\_supported | The OP does not support use of the request\_uri parameter |
| registration\_not\_supported | The OP does not support use of the registration parameter |

A non-normative example follows:

HTTP/1.1 302 Found

Location: https://client.example.org/cb?

error=invalid\_request

&error\_description=Unsupported%20response\_type%20value

&state=af0ifjsldkj

## Token Endpoint

### Token Request

References:

* OpenID Connect Core Specification [1], s3.1.3.1 & s12
* International Government Assurance Profile (iGov) for OpenID Connect [2]
* RFC6749 - The OAuth 2.0 Authorization Framework [6], s4.1.3 & s6

A Token request is used to obtain an Access Token and an ID Token. The Partner Service (client) sends a Token Request to the Token Endpoint to obtain a Token Response.

Confidential Clients MUST authenticate to the Token Endpoint using the authentication method registered for its client\_id (see section 7.2.1).

The Client sends the parameters to the Token Endpoint using the HTTP POST method and the Form Serialization(HTTP Content-Type application/x-www-form-urlencoded) – the request must be sent using TLS v1.2 or above.

Table 6: Token Request Parameters

|  |  |  |
| --- | --- | --- |
| Parameter | Req? | Description |
| grant\_type | Mand | Value MUST be set to "authorization\_code" OR “refresh\_token” |
| code | Cond | The authorization code previously received from the Platform. MUST be present when grant\_type is “authorization\_code” |
| redirect\_uri | Cond | Required when grant\_type is “authorization\_code” – MUST match the "redirect\_uri" parameter that was included in the authorization request. |
| client\_id | Cond | Optional for Confidential Clients |
| client\_assertion\_type | Mand | Mandatory for Confidential Clients  “urn:ietf:params:oauth:client-assertion-type:jwt-bearer” |
| client\_assertion | Mand | Mandatory for Confidential Clients  A signed JWT presented for Client Authentication, as per  <https://tools.ietf.org/html/rfc7523>  (Note the “iss” and “sub” claim MUST equal the client\_id and the “aud” MUST contain the token endpoint URL)  The assertion MUST be signed using the private key of the client, agreed during onboarding |
| refresh\_token | Cond | The refresh token previously issued to the client. MUST be present when grant\_type is “refresh\_token” |
| scope | Opt | Only applicable for grant\_type of “refresh\_token”. The scope of the access request. The requested scope MUST NOT include any scope not originally granted by the originating “authorization\_code” grant. If omitted is treated as equal to the scope originally granted by the “authorization\_code” grant. |

A non-normative example follows:

POST /token HTTP/1.1

Host: auth.login.nhs.uk

Content-Type: application/x-www-form-urlencoded

grant\_type=authorization\_code

&code=SplxlOBeZQQYbYS6WxSbIA

&redirect\_uri=https%3A%2F%2Fclient.example.org%2Fcb

&client\_assertion\_type=urn%3Aietf%3Aparams%3Aoauth%3Aclient-assertion-type%3Ajwt-bearer

&client\_assertion=PHNhbWxwOl ... ZT

### Token Request Validation

References:

* OpenID Connect Core Specification [1], s3.1.3.2
* RFC6749 - The OAuth 2.0 Authorization Framework [6], s6

The Token Request is validated as described in the OpenID Connect Core Specification [1], section 3.1.3.2 (for grant\_type “authorization\_code”) or as described in The OAuth 2.0 Authorization Framework [6] (for grant\_type “refresh\_token”). The client\_assertion is validated as per <https://tools.ietf.org/html/rfc7523> for Client Authentication for all requests.

### Token Response

#### Successful Response

References:

* OpenID Connect Core Specification [1], section 3.1.3.3 & s12.2
* RFC6749 - The OAuth 2.0 Authorization Framework [6], s4.1.4 & s6

After receiving and validating a valid and authorised Token request from the client, the Token Endpoint returns a response which includes an ID Token and an Access Token. The response uses the “application/json” media type.

Table 7: Token Response: HTTP headers & values

|  |  |
| --- | --- |
| Header Name | Value |
| Cache-Control | no-store |
| Pragma | no-cache |

Table 8: Token Response

| Parameter | Req? | Description |
| --- | --- | --- |
| access\_token | Mand | Signed JWT which encodes the Access Token, see sections 4.1 and 4.3 |
| token\_type | Mand | Must be value “bearer” |
| refresh\_token | Opt | The refresh token which can be used to obtain new access tokens as described in section 3.5.1 |
| expires\_in | Opt | Recommended.  The lifetime in seconds of the access token. If omitted, the Platform SHOULD provide the expiration time via other means or document the default value |
| scope | Cond | OPTIONAL, if identical to the scope requested by the client;  otherwise, REQUIRED |
| id\_token | Cond | Signed JWT which encodes the ID Token – see sections 4.1 and 4.2. Only issued for grant\_type “authorization\_code” |

A non-normative example follows:

HTTP/1.1 200 OK

Content-Type: application/json

Cache-Control: no-store

Pragma: no-cache

{

"access\_token": "SlAV32hkKG",

"refresh\_token": "dS4tfWD",

"token\_type": "Bearer",

"expires\_in": 3600,

"id\_token": "eyJhbGciOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ.ewogImlzc

yI6ICJodHRwOi8vc2VydmVyLmV4YW1wbGUuY29tIiwKICJzdWIiOiAiMjQ4Mjg5

NzYxMDAxIiwKICJhdWQiOiAiczZCaGRSa3F0MyIsCiAibm9uY2UiOiAibi0wUzZ

fV3pBMk1qIiwKICJleHAiOiAxMzExMjgxOTcwLAogImlhdCI6IDEzMTEyODA5Nz

AKfQ.ggW8hZ1EuVLuxNuuIJKX\_V8a\_OMXzR0EHR9R6jgdqrOOF4daGU96Sr\_P6q

Jp6IcmD3HP99Obi1PRs-cwh3LO-p146waJ8IhehcwL7F09JdijmBqkvPeB2T9CJ

NqeGpe-gccMg4vfKjkM8FcGvnzZUN4\_KSP0aAp1tOJ1zZwgjxqGByKHiOtX7Tpd

QyHE5lcMiKPXfEIQILVq0pc\_E2DzL7emopWoaoZTF\_m0\_N0YzFC6g6EJbOEoRoS

K5hoDalrcvRYLSrQAZZKflyuVCyixEoV9GfNQC3\_osjzw2PAithfubEEBLuVVk4

XUVrWOLrLl0nx7RkKU8NXNHq-rvKMzqg"

}

#### Error Response

References:

* OpenID Connect Core Specification [1], s3.1.3.4
* RFC6749 - The OAuth 2.0 Authorization Framework [6], s5.2

If the Token Request is invalid or unauthorized, the Platform constructs the error response. The parameters of the Token Error Response are defined as in Section 5.2 of OAuth 2.0 [RFC6749]. The HTTP response body uses the application/json media type with HTTP response code of 400.

The following is a non-normative example Token Error Response:

HTTP/1.1 400 Bad Request

Content-Type: application/json

Cache-Control: no-store

Pragma: no-cache

{

"error": "invalid\_request"

}

Table 9: Token Error Response

|  |  |  |
| --- | --- | --- |
| Parameter | Req? | Description |
| error | Mand | Error code |
| error\_description | Opt | Human-readable ASCII encoded text description of the error |
| error\_uri | Opt | URI of a web page that includes additional information about the error |

Table 10: Error Codes for Token Error Response

|  |  |
| --- | --- |
| Code | Description |
| OAuth 2.0 error codes | |
| invalid\_request | The request is missing a required parameter, includes an unsupported parameter value (other than grant type), repeats a parameter, includes multiple credentials, utilizes more than one mechanism for authenticating the client, or is otherwise malformed. |
| invalid\_client | Client authentication failed (e.g., unknown client, no client authentication included, or unsupported authentication method). |
| invalid\_grant | The provided authorization grant (e.g., authorization code, resource owner credentials) or refresh token is invalid, expired, revoked, does not match the redirection URI used in the authorization request, or was issued to another client. |
| unauthorized\_client | The authenticated client is not authorized to use this authorization grant type. |
| unsupported\_grant\_type | The authorization grant type is not supported by the Platform. |
| invalid\_scope | The requested scope is invalid, unknown, malformed, or exceeds the scope granted by the resource owner |

## UserInfo Endpoint

References:

* OpenID Connect Core Specification [1], s5.3

The UserInfo Endpoint is an OAuth 2.0 Protected Resource that returns Claims about the authenticated End-User. To obtain the requested Claims about the End-User, the Client makes a request to the UserInfo Endpoint using an Access Token obtained through OpenID Connect Authentication. These Claims are represented by a JSON object that contains a collection of name and value pairs for the Claims.

### UserInfo Request

References:

* OpenID Connect Core Specification [1], s5.3.1
* RFC6750: OAuth 2.0 Bearer Token Usage [7], s2

The Client sends the UserInfo Request using either HTTP GET (recommended) or HTTP POST.

The Access Token obtained from an OpenID Connect Authentication Request MUST be sent as a Bearer Token using the Authorization header field, per Section 2 of OAuth 2.0 Bearer Token Usage.

The following is a non-normative example of a UserInfo Request:

GET /userinfo HTTP/1.1

Host: auth.login.nhs.uk

Authorization: Bearer SlAV32hkKG

### UserInfo Response

#### Successful Response

Reference:

* OpenID Connect Core Specification [1], s5.3.2

The UserInfo Claims SHALL be returned as the members of a JSON object. If a Claim is not returned, then that Claim Name SHALL be omitted from the JSON object representing the Claims; it SHALL NOT be present with a null or empty string value.

Claims will ONLY be returned when a user has consented to the information being released to the client.

References:

* OpenID Connect Core Specification [1], section 5.1

The NHS login Service supports standard and additional claims as follows:

Table 11: NHS login Supported Claims

| Claim | Supported | Description |
| --- | --- | --- |
| **OpenID Connect Standard Claims** | | |
| iss | Y | Issuer Identifier for the Issuer of the response |
| aud | Y | The Partner Service identifier |
| sub | Y | Subject - Identifier for the End-User at the Issuer. |
| family\_name | Y | Surname(s) or last name(s) of the End-User |
| given\_name | Y | First name(s) of the End-User.  This information will only be returned where the user’s identity has been verified AND the profile\_extended scope is requested |
| email | Y | End-User's preferred e-mail address  Present if the email scope was present in the request |
| email\_verified | Y | True if the End-User's e-mail address has been verified; otherwise false  Present if the email scope was present in the request |
| phone\_number | Y | End-User's preferred phone number  Present if the phone scope was present in the request |
| phone\_number\_verified | Y | True if the End-User's phone number has been verified; otherwise false  Present if the phone scope was present in the request |
| birthdate | Y | End-User’s date of birth in ISO8601‑2004 format:  YYYY-MM-DD |
| **NHS login Additional Claims** | | |
| nhs\_number | Y | A string containing the End User’s NHS Number – this is a 10 digit string |
| gp\_integration\_credentials | Y | A json object containing the end user’s GP integration credentials  These will only be returned where the user’s identity has been verified AND the gp\_integration\_credentials scope is requested  {  gp\_user\_id: “32498239048-3248734”,  gp\_linkage\_key: “dfje2rkjdfkjdfm”,  gp\_ods\_code: “A12344”  } |
| gp\_registration\_details | Y | A json object containing information on the End-User’s registered General Practice as held in NHS Personal Demographics Service.  This information will only be returned where the user’s identity has been verified AND the gp\_registration\_details scope is requested  {  gp\_ods\_code: “A12344”  } |
| client\_user\_metadata | Y | A string containing **either** a randomly-generated (by NHS login) number which has been encrypted using PKCS1\_OAEP cipher with the RP’s public key and then base64URL encoded **or** a value set by the RP. The value contained in this claim is opaque to NHS login and RP may optionally update this value – see section 3.6.3  If no value is held for the RP then this claim is not returned. |
| identity\_proofing\_level | Y | A string containing the End User’s identity proofing level. It will contain one of the values defined in section [5.1.1 -Verification of Identity Levels](#_Verification_of_Identity) and Table 18 – e.g. “P0”, “P9” |

The sub Claim in the UserInfo Response MUST be verified to exactly match the sub Claim in the ID Token; if they do not match, the UserInfo Response values MUST NOT be used.

The UserInfo response MAY contain other Claims. Any Claims used that are not understood MUST be ignored by the RP.

Upon receipt of the UserInfo Request, the UserInfo Endpoint SHALL return the JSON Serialization of the UserInfo Response in the HTTP response. The content-type of the HTTP response SHALL be “application/json; the response body SHALL be encoded using UTF-8.

The UserInfo Response will not be signed and/or encrypted.

The following is a non-normative example of a UserInfo Response:

HTTP/1.1 200 OK

Content-Type: application/json

{

“iss”: “https://login.nhs.uk”,

“sub”: “24400320-234545-234241-111”,

“aud”: “s6BhdRkqt3”,

“email”: “janedoe@example.com”,

“email\_verified”: true,

“phone\_number”: “01234567891”,

“phone\_number\_verified”: true,

“nhs\_number”: “8527685222”,

“birthdate”: 2001-12-30,

“family\_name”: “Doe”,

“identity\_proofing\_level”: “P9”,

“gp\_integration\_credentials”: {

gp\_user\_id: “32498239048-3248734”,

gp\_linkage\_key: “dfje2rkjdfkjdfm”,

gp\_ods\_code: “A12344”

},

“client\_user\_metadata”: “U2e3rsdjwd…==”

}

#### Error Response

References:

* OpenID Connect Core Specification [1], s5.3.3
* RFC6750: OAuth 2.0 Bearer Token Usage [7], s3

When an error condition occurs, the UserInfo Endpoint returns an Error Response which will include JSON keys as defined in Table 12. As defined in the OpenID Connect Core Specification, errors relating to Bearer Token Usage are returned using the HTTP “WWW-Authenticate” response header field as per Section 3 of OAuth 2.0 Bearer Token Usage. Errors not related to Bearer Token Usage omit the “WWW-Authenticate” response header field and return error details in the response body.

Table 12: UserInfo Error Response

| JSON Key | Req? | Description |
| --- | --- | --- |
| error | Mand | Error code, see Table 13 for permitted values |
| error\_description | Opt | Human-readable ASCII encoded text description of the error |

Table 13: Error Codes for UserInfo Error Response

| Code | Description |
| --- | --- |
| **Oauth 2.0 error codes** | |
| invalid\_request | The request is missing a required parameter, includes an unsupported parameter or parameter value, repeats the same parameter, uses more than one method for including an access token, or is otherwise malformed. The resource server SHOULD respond with the HTTP 400 (Bad Request) status code. |
| invalid\_token | The access token provided is expired, revoked, malformed, or invalid for other reasons. The resource SHOULD respond with the HTTP 401 (Unauthorized) status code. The client MAY request a new access token and retry the protected resource request. |
| insufficient\_scope | The request requires higher privileges than provided by the access token. The resource server SHOULD respond with the HTTP 403 (Forbidden) status code and MAY include the “scope” attribute with the scope necessary to access the protected resource |

The following is a non-normative example of a UserInfo Error Response relating to Bearer Token Usage:

HTTP/1.1 401 Unauthorized

WWW-Authenticate: error=”invalid\_token”,

error\_description=”The access token expired”

The following is a non-normative example of a UserInfo Error Response not relating to Bearer Token Usage:

HTTP/1.1 400 Bad Request

Content-Type: application/json

{

“error”: “invalid\_request”,

“error\_description”: ”Invalid parameter content”

}

### UserInfo Update Request

NHS login extends the OpenID Connect Specification and allows a client to request the update of the client\_user\_metadata claim via HTTP PATCH.

The Access Token obtained from an OpenID Connect Authentication Request MUST be sent as a Bearer Token using the Authorization header field, per Section 2 of OAuth 2.0 Bearer Token Usage.

The Client sends the parameters to the UserInfo Endpoint using the HTTP PATCH method and a Content-Type of ‘application/json’ – the request must be sent using TLS v1.2 or above.

Table 14 - UserInfo Update Request Parameters

|  |  |  |
| --- | --- | --- |
| Json Key | Req? | Description |
| client\_user\_metadata | Opt | New value for the client\_user\_metadata claim. The maximum length is 1024 characters. Setting the value to an empty string (i.e. client\_user\_metadata=) will result in NHS login clearing the value. Setting the value to “auto” will result in NHS login re-generating a new value as described in Table 11. |

As the client\_user\_metadata value is opaque to NHS login no specific validation will be undertaken other than to ensure a maximum length of 1024 characters.

The following is a non-normative example of a UserInfo Update Request:

PATCH /userinfo HTTP/1.1

Host: auth.login.nhs.uk

Authorization: Bearer SlAV32hkKG

Content-Type: application/json

{‘client\_user\_metadata’: ‘ZT&PHNhbWxwOl ... ZT’}

### UserInfo Update Response

A successful UserInfo Update Response will be an HTTP 204 response with no response body. Error responses will be as described in section 3.6.2.2

# Tokens

An ID Token is a security token that contains Claims about the Authentication of an End-User by the Platform, when using a Client and potentially other requested Claims. The Access Token is a credential that can be used by an application to access an API – initially the UserInfo endpoint.

The NHS login Platform uses signed JSON Web Tokens (JWTs) for ID Tokens and Access Tokens.

## JWT Header

References:

* RFC7519 – JSON Web Token (JWT) [4]
* RFC7515 – JSON Web Signature (JWS) [8]

The JWT header will contain the following claims:

Table 15: JWT Header

| Claim | Req? | Name | Description |
| --- | --- | --- | --- |
| alg | Mand | Algorithm used for signing the JWT | “RS512” – RSASSA-PKCS1-v1\_5 with the SHA-512 hash algorithm |
| typ | Mand | Type | “JWT” |
| jku | n/a | JWK Set URL | Not supported, keys are communicated outside the JWT, see Ref [1], s2 |
| jwk | n/a | JSON Web Key | Not supported, keys are communicated outside the JWT, see Ref [1], s2 |
| x5u | n/a | X.509 URL | Not supported, keys are communicated outside the JWT, see Ref [1], s2 |
| x5c | n/a | X.509 Certificate Chain | Not supported, keys are communicated outside the JWT, see Ref [1], s2 |

The JWT header MAY contain other Claims. Any Claims used that are not understood MUST be ignored by the RP.

## ID Token Payload

References:

* OpenID Connect Core Specification [1], section 2

The following Claims are used within the ID Token for all Oauth 2.0 flows used by OpenID Connect:

Table 16: id\_token claims

| Claim | Req? | Name | Description |
| --- | --- | --- | --- |
| **Standard Claims** | | | |
| iss | Mand | Issuer Identifier for the Issuer of the response | The iss value is a case sensitive URL using the https scheme that contains scheme, host, and optionally, port number and path components and no query or fragment components. |
| sub | Mand | Subject Identifier | A locally unique and never reassigned identifier within the Issuer for the End-User, which is intended to be consumed by the Client, e.g., 24400320 or AitOawmwtWwcT0k51BayewNvutrJUqsvl6qs7A4  It SHALL NOT exceed 255 ASCII characters in length.  The sub value is a case sensitive string |
| aud | Mand | Audience(s) that this ID Token is intended for | It SHALL contain the Oauth 2.0 client\_id of the Partner Services as an audience value. It MAY also contain identifiers for other audiences. In the general case, the aud value is an array of case sensitive strings. In the common special case when there is one audience, the aud value SHALL be a single case sensitive string |
| exp | Mand | Expiration time on or after which the ID Token MUST NOT be accepted for processing | The processing of this parameter requires that the current date/time MUST be before the expiration date/time listed in the value. Implementers MAY provide for some small leeway, usually no more than a few minutes, to account for clock skew. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. See RFC 3339 [RFC3339] for details regarding date/times in general and UTC in particular |
| iat | Mand | Time at which the JWT was issued | Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. See RFC 3339 [9] for details regarding date/times in general and UTC in particular |
| jti | Mand | JWT unique identifier | Value is unique to each token created by the issuer |
| auth\_time | Cond | Time when the End-User authentication occurred | Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.  When a max\_age request is made or when auth\_time is requested as an Essential Claim, then this Claim is REQUIRED; otherwise, its inclusion is OPTIONAL. (The auth\_time Claim semantically corresponds to the OpenID 2.0 PAPE [10] auth\_time response parameter.) |
| nonce | Cond | String value used to associate a Client session with an ID Token, and to mitigate replay attacks | The value is passed through unmodified from the Authentication Request to the ID Token.  If present in the ID Token, Clients MUST verify that the nonce Claim Value is equal to the value of the nonce parameter sent in the Authentication Request.  If present in the Authentication Request, the Platform MUST include a nonce Claim in the ID Token with the Claim Value being the nonce value sent in the Authentication Request.  The Platform will perform no other processing on nonce values used.  The nonce value is a case sensitive string |
| acr | n/a | Authentication Context Class Reference | Not Supported – superseded by vot |
| amr | n/a | Authentication Methods References | Not currently supported |
| azp | n/a | Authorized party – the party to which the ID Token was issued | Not currently supported |
| vot | Mand | Vectors of Trust | The level to which the user’s identity has been verified.  See s5.1 for values  Returned values should be verified and must match to one or more values of the vtr (vector of trust request) in the authorize request (see section [3.4.1](#_Authentication_Request)). |
| vtm | Mand | Vector Trust Mark | https URI of the vtm claim  See s5.1.5 |
| family\_name | Cond | Family Name | Surname(s) or last name(s) of the End-User |
| birthdate | Cond | Birthdate | If the user’s date of birth is available, it is returned in ISO8601‑2004 format, YYYY-MM-DD |
| **NHS login extensions** | | | |
| nhs\_number | Cond | NHS Number | If the user’s NHS Number is known, then it SHALL be included (subject to the profile scope being requested). Will be represented as a 10-digit string. |
| identity\_proofing\_level | Cond | Identity Proofing Level | User’s identity proofing level shall be included (subject to the profile scope being requested). It will contain one of the values defined in section [5.1.1 -Verification of Identity Levels](#_Verification_of_Identity) and Table 18 – e.g. “P0”, “P9”. |

ID Tokens MAY contain other Claims. Any Claims used that are not understood MUST be ignored by the RP.

The following is a non-normative example of the set of Claims (the JWT Claims Set) in an ID Token:

{

“iss”: “ <https://auth.login.nhs.uk>”,

“sub”: “24400320”,

“aud”: “s6BhdRkqt3”,

“nonce”: “n-0S6\_WzA2Mj”,

“exp”: 1311281970,

“iat”: 1311280970,

“auth\_time”: 1311280969,

“vot”: “P9.Ca.Cc”,

“vtm”: “https://auth.login.nhs.uk/trustmark/auth.login.nhs.uk”

“nhs\_number”: “4447685222”,

“birthdate”: “2001-12-30”,

“family\_name”: “Johnson”,

“identity\_proofing\_level”: “P9”

}

## Access Token Payload

References:

* OpenID Connect Core Specification [1]
* Access Tokens and Audit (JWT) [11]

Table 17: access\_token claims

| Claim | Req? | Name | Description |
| --- | --- | --- | --- |
| iss | Mand | Issuer Identifier for the Issuer of the response | The iss value is a case sensitive URL using the https scheme that contains scheme, host, and optionally, port number and path components and no query or fragment components. |
| sub | Mand | Subject Identifier | A locally unique and never reassigned identifier within the Issuer for the End-User, which is intended to be consumed by the Client, e.g., 24400320 or AitOawmwtWwcT0k51BayewNvutrJUqsvl6qs7A4. It SHALL NOT exceed 255 ASCII characters in length.  The sub value is a case sensitive string |
| aud | Mand | Audience(s) that this ID Token is intended for | It SHALL contain the Oauth 2.0 client\_id of the Partner Service as an audience value. It MAY also contain identifiers for other audiences. In the general case, the aud value is an array of case sensitive strings. In the common special case when there is one audience, the aud value SHALL be a single case sensitive string |
| exp | Mand | Expiration time on or after which the ID Token MUST NOT be accepted for processing | The processing of this parameter requires that the current date/time MUST be before the expiration date/time listed in the value. Implementers MAY provide for some small leeway, usually no more than a few minutes, to account for clock skew. Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. See RFC 3339 [RFC3339] for details regarding date/times in general and UTC in particular |
| iat | Mand | Time at which the JWT was issued | Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time. See RFC 3339 [RFC3339] for details regarding date/times in general and UTC in particular |
| jti | Mand | JWT unique identifier | Value is unique to each token created by the issuer |
| scope | Mand | Scopes that the access\_token provides access to | A space-separated list of scopes for which the token is issued. This list will be scopes requested in the authentication request (see s3.4.1) or a subset of them |
| auth\_time | Cond | Time when the End-User authentication occurred | Its value is a JSON number representing the number of seconds from 1970-01-01T0:0:0Z as measured in UTC until the date/time.  When a max\_age request is made or when auth\_time is requested as an Essential Claim, then this Claim is REQUIRED; otherwise, its inclusion is OPTIONAL. (The auth\_time Claim semantically corresponds to the OpenID 2.0 PAPE [10] auth\_time response parameter.) |
| acr | n/a | Authentication Context Class Reference | Not supported – superseded by vot |
| vot | Mand | Vectors of Trust | The level to which the user’s identity has been verified.  See s5.1 for values  Returned values should be verified and must match to one or more values of the vtr (vector of trust request) in the authorize request (see section [3.4.1](#_Authentication_Request)). |
| vtm | Mand | Vector Trust Mark | https URI of the vtm claim  See s5.1.5 |
| **NHS login extensions** | | | | |
| nhs\_number | Cond | NHS Number | If the user’s NHS Number is known, then it SHALL be included (subject to the profile scope being requested). Will be represented as a 10-digit string. |
| **Spine Core extensions** | | | | |
| reason\_for\_request | Opt | The identified the purpose for which the request is being made. | Purpose for which access is being requested. Will contain “patientaccess”. See [11] |
| requesting\_system | Opt | Identifier for the system or device making the request | This is an identifier for the deployed client system that has been authorised to make API calls. In the case of Spine-enabled clients (or those using the SSP to broker API calls), this will be a Spine Accredited System ID (ASID)  The naming system prefix for the ASID will be <https://fhir.nhs.uk/Id/accredited-system>  See [11] |
| requesting\_patient | Opt | NHS Number | If this authorisation relates to a citizen, this attribute will hold the NHS Number of the citizen  The naming system prefix for the NHS Number will be <http://fhir.nhs.net/Id/nhs-number>  See [11] |

Access Tokens MAY contain other Claims. Any Claims used that are not understood MUST be ignored by the RP.

## JOSE Signing

References:

* RFC7519 – JSON Web Token (JWT) [4]
* RFC7515 – JSON Web Signature (JWS) [8]

The JWT will be signed using the RSASSA-PKCS1-v1\_5 with the SHA-512 hash algorithm (“RS512”)

# Data View

## Vectors of Trust

**References**

* DCB3051 Identity Verification and Authentication Standard for Digital Health and Care Services [12]
* International Government Assurance Profile (iGov) [2]
* RFC8485 - Vectors of Trust [3]

NHS login supports Vectors of Trust for Identification Verification and Authentication. Credential Management and Assertion Presentation are out of scope

### Verification of Identity Levels

NHS login supports the following levels of identification verification. These are based upon ‘DCB3051 Identity Verification and Authentication Standard for Digital Health and Care Services’ [12].

Table 18: NHS login Verification Levels

| Value | Description |
| --- | --- |
| P0 | No Identity proofing  This maps to ‘Verification – Low’ within DCB3051 |
| P5 | Knowledge-based verification ONLY  This maps to ‘Verification – Medium’ within DCB3051 |
| P9 | Physical comparison  This maps to ‘Verification – High’ within DCB3051 |

### Authentication Credentials

NHS login supports the following types of authentication credentials. Note that the credential component may occur more than once

Table 19: NHS login Authentication Context Class

| Value | Description |
| --- | --- |
| Cp | Password-based  The user is in possession of a secret (for example a password, PIN, etc) belonging to the legitimate account holder. |
| Cd | Registered Device  The user is in possession of a device which has previously been associated with their account – delivery/use of the device is by identifiers only |
| Ck | Shared Cryptographic key within a Registered Device  The user is in possession of a device which has previously been associated with their account – delivery/use of the device is by a **shared** key |
| Cm | Asymmetric Cryptographic key within a Registered Device  The user is in possession of a device which has previously been associated with their account – delivery/use of the device is by cryptographic proof of key possession using **asymmetric** key, such as a FIDO-compliant device |

### Requesting Vectors of Trust values

Vector of Trust (VoT) is a combination of - Identity Verification (defined in section [5.1.1 -Verification of Identity Levels](#_Verification_of_Identity)) and Authentication (defined in section [5.1.2 – Authentication Credentials](#_Authentication_Credentials)) levels.

The client MAY request a set of acceptable VoT values with the "vtr" (vector of trust request) claim request as part of the Authentication Request. The value of this field is an array of JSON strings, each string identifying an acceptable set of vector components. The component values within each vector are ANDed together while the separate vectors are ORed together. For example, a list of vectors in the form ‘["P9.Cp.Cd ", "P9.Ck"]’ is stating that either the full set of "P9 AND Cp AND Cd" simultaneously OR the full set of "P9 AND Ck" simultaneously are acceptable for this transaction.

The requested set of VoT, SHOULD only contain one value for Identity Verification component, combined with one or more values of the Authentication component.

Client SHOULD only request the lowest level of acceptable Identity Verification, within the VoT set.Vector request values MAY omit components, indicating that any value is acceptable for that component category, including omission of that component in the response vector.

### Returning Vectors of Trust values

The vector is sent as a string within the "vot" (vector of trust) claim in the ID and Access tokens. The trustmark that applies to this vector SHALL be sent as an HTTPS URL in the "vtm" (vector trust mark) claim to provide context to the vector.

Returned "vot" (vector of trust) claim values in the ID and Access tokens should be verified and MUST match to one or more values of the requesting vtr (vector of trust request) in the authorize request (see section [3.4.1](#_Authentication_Request)).

For example, part of the body of an ID token claiming "Medium Assurance Identity, authenticated user via password and registered device identifier" could look like this JSON object payload of the ID token.

{

…

…

"iss": " https://auth.login.nhs.uk/",

"vot": "P5.Cp.Cd",

"vtm": <https://auth.login.nhs.uk/trustmark/login.nhs.uk>

…

…

}

The body of the token is signed using JOSE, as per the OpenID Connect specification. By putting the "vot" and "vtm" values inside the token, the vector and its context are strongly bound to the federated credential represented by the token.

### Trustmark

**References**

* RFC8485 - Vectors of Trust [3], section 5

The Trustmark provides a list of claims that NHS login supports. This enables the client to verify which components of a trust framework NHS login supports and hence their trustworthiness

The NHS login Trustmark will be self-hosted.

The following is a non-normative example a Trustmark based on s5.1.3 and 5.1.4 above:

{

"idp": "https://auth.login.nhs.uk/",

"trustmark\_provider": "https://auth.login.nhs.uk/",

"P": ["P0", "P3", "P5", "P6", "P7", "P9"],

"C": ["Cp", "Cd", "Ck", "Cm"]

}

### Profiles

* Aligning the above to the standard DCB3051 Identity Verification and Authentication Standard for Digital Health and Care Services [12] brings a standard set of vectors supported by the NHS login Service, described in the table below

| Vector | Description |
| --- | --- |
| “P0.Cp” | Low Identity verification, user authenticated using password |
| “P0.Cp.Cd” | Low Identity verification, user authenticated using password and enrolled device (2FA) |
| “P0.Cp.Ck” | Low Identity verification, user authenticated using password and shared key within a device (2FA) |
| “P0.Cm” | Low Identity verification, user authenticated via asymmetric key within a device (2FA) (for example, FIDO UAF authentication) |
| “P5.Cp” | Medium Identity verification, user authenticated using password |
| “P5.Cp.Cd” | Medium Identity verification, user authenticated using password and enrolled device (2FA) |
| “P5.Cp.Ck” | Medium Identity verification, user authenticated using password and shared key within a device (2FA) |
| “P5.Cm” | Medium Identity verification, user authenticated via asymmetric key within a device (2FA) (for example, FIDO UAF authentication) |
| “P9.Cp” | High Identity verification, user authenticated using password |
| “P9.Cp.Cd” | High Identity verification, user authenticated using password and enrolled device (2FA) |
| “P9.Cp.Ck” | High Identity verification, user authenticated using password and shared key within a device (2FA) |
| “P9.Cm” | High Identity verification, user authenticated via asymmetric key within a device (2FA) (for example, FIDO UAF authentication) |

#### Example 1 – Partner Service provides access to sensitive data

This example maps onto archetypes A1, A4, A5, A6, A7 within the standard ‘DCB3051 Identity Verification and Authentication Standard for Digital Health and Care Services’ [12]

The service requires High Verification and High Authentication. The service sends the following as acceptable Vectors in the ‘vtr’ parameter:

“[“P9.Cp.Cd”,“P9.Cp.Ck”,“P9.Cm”]”

#### Example 2 – Partner Service provides access to basic data (not sensitive)

This example maps onto archetype A3 within the standard ‘DCB3051 Identity Verification and Authentication Standard for Digital Health and Care Services’ [12]

The service requires Medium Verification and High Authentication, with SSO allowed. The service sends the following as acceptable Vectors in the ‘vtr’ parameter:

“[“P5.Cp.Cd”, “P5.Cp.Ck”,“P5.Cm”]”

#### Example 3 – Partner Service provides access to both basic data (not sensitive) and sensitive data

This example maps onto a service offering multiple features, of which some require basic user data, and some require sensitive user data.

Service requests NHS login for Medium Verification (P5) and High Authentication for a user. The service sends the following as acceptable Vectors in the ‘vtr’ parameter:

“[“P5.Cp.Cd”,“P5.Cp.Ck”,“P5.Cm”]”

Successfully authenticated user can either be P5 or P9 in terms of identity verification.

For a P9 user (high identity verification) the service can offer all the functionalities, as the user meets service requirements of High Verification and High Authentication

For a P5 user (medium identity verification) the service offers basic features at the user’s initial login as the user meets service requirements of Medium Verification and High Authentication only.

When a P5 user, attempts to access other features supported by sensitive data then the service requires High Verification and High Authentication, with SSO allowed. The service sends the following as acceptable Vectors in the ‘vtr’ parameter:

“[“P9.Cp.Cd”,“P9.Cp.Ck”,“P9.Cm”]”

The user is prompted to undertake a verification step-up journey to take the user verification level from Medium to High.

#### Single Sign-on

If the user’s current sign-on session (if any), does not meet the requested Vectors of Trust (vtr) in the incoming request, then the user will be required to sign-in, in order to meet the requested vectors.

Also note that Single sign-on behaviour, or to refuse SSO-behaviour, can be controlled using the ‘prompt’ parameter on the initial authorisation request – see section 3.4.1.

# Password-less Authentication using FIDO UAF

References:

* FIDO Universal Authentication Framework version 1.0 [15]
* FIDO UAF Registry of Predefined Values version 1.0 [16]
* FIDO UAF Authenticator Commands version 1.0 [17]
* OAuth 2.0 Bearer Token Usage [7]

This section describes how the NHS Digital NHS login Platform implements password-less authentication using a combination of FIDO Universal Authentication Framework and the OpenID Connect Authorization Code Flow.

The NHS login Platform supports the FIDO UAF Protocol version 1.0.

## Overview

The objective of utilising FIDO UAF in combination with the OpenID Connect Authorization Code flow is to enable native app clients to offer an enhanced login experience to users, allowing them to utilise device-based biometric authentication (e.g. fingerprint).

NHS login exposes four endpoints to support password-less authentication via FIDO UAF. These are :

* regRequest – enables a client to request a FIDO UAF Registration Request Message from the NHS login platform. The access token retrieved by the client during user authentication MUST be presented in the Authorization HTTP Header as a Bearer token as per RFC6750
* regResponse – enables a client to POST a FIDO UAF Registration Response Message to the NHS login platform.
* dereg – enables a client to POST a FIDO UAF Deregistration Request Message to the NHS login platform specifying a set of keys to delete. The NHS login platform will remove the relevant user key(s) and return a FIDO UAF Deregistration Request Message for processing by the FIDO client. The access token retrieved by the client during user authentication MUST be presented in the Authorization HTTP Header as a Bearer token as per RFC6750
* authRequest – enables a client to request a FIDO UAF Authentication Request Message from the NHS login platform. The client processes the FIDO UAF Authentication Request Message and creates a FIDO UAF Authentication Response Message which the client then base64 URL encodes and provides to the NHS login platform as an authentication request parameter (fido\_auth\_response) as described in section 3.4.1

The endpoints are published in the /.well-known/openid-configuration document, under the keys "fido\_uaf\_registration\_request\_endpoint", "fido\_uaf\_registration\_response\_endpoint", "fido\_uaf\_deregistration\_endpoint" and "fido\_uaf\_authentication\_request\_endpoint"

## FIDO UAF registration flow

Figure 4 below is a non-normative illustration of the FIDO UAF Registration flow (taken from the FIDO UAF Architectural Overview document [15]). The process is initiated after user authentication using the standard NHS login OIDC Authorization Code Flow (as per section [3.1](#_Authorization_Code_Flow) of this specification).

The client initiates registration by invoking the regRequest endpoint of the NHS login platform (1). The NHS login platform generates a FIDO UAF Registration Request Message and returns this to the client, which passes it to the FIDO Client on the device for processing (2). The FIDO Client interacts with the FIDO Authenticators on the user’s device (3) and creates a FIDO UAF Registration Response Message corresponding to the original FIDO UAF Registration Request Message. The FIDO Client passes this message back to the client which sends it to the regResponse endpoint of the NHS login service (4). The NHS login platform validates the details held within the FIDO UAF Registration Response Message and stores the relevant details (5). FIDO UAF registration is complete.

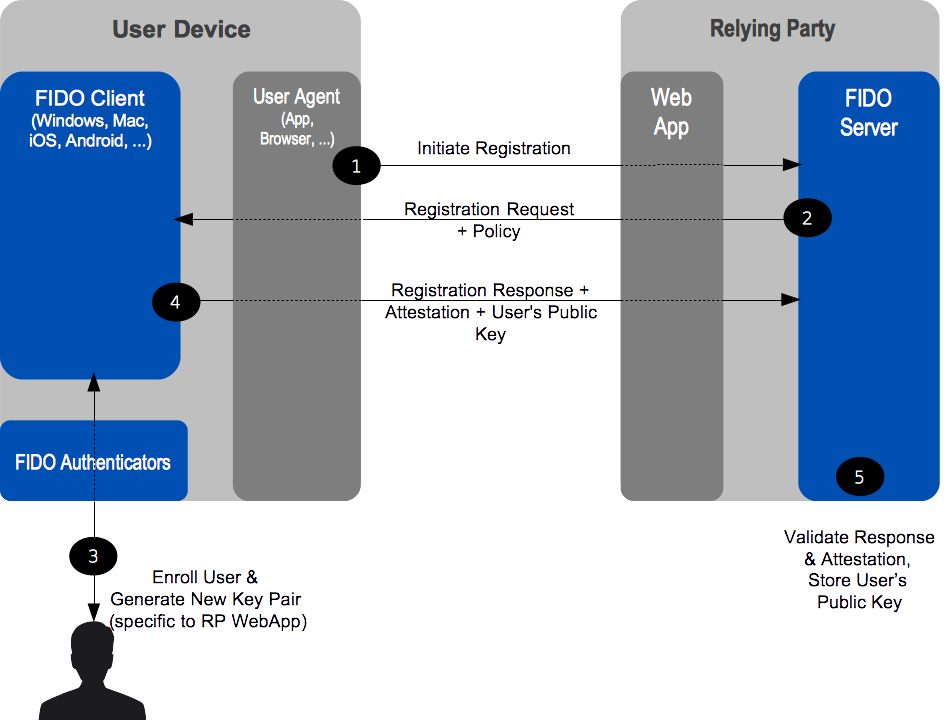


Figure 4 - FIDO UAF Registration flow

### Registration Request endpoint

#### Request Syntax

The Client sends the request for a FIDO UAF Registration Request Message using HTTP GET.

The Access Token obtained from an OpenID Connect Authentication Request MUST be sent as a Bearer Token using the Authorization header field, per Section 2 of OAuth 2.0 Bearer Token Usage.

The following is a non-normative example of a regRequest Request:

GET /regRequest HTTP/1.1

Host: uaf.prod.signin.nhs.uk

Authorization: Bearer SlAV…32hkKG

#### Registration Request Validation

The access token presented in the request is validated prior to generating a FIDO UAF Registration Response Message as per section 3.4.6.1 of the FIDO UAF Protocol Specification.

#### Registration Request Response

The response from the Registration Request endpoint will be a FIDO UAF Registration Request Message as per section 3.4.1 of the FIDO UAF Protocol Specification.

The following is a non-normative[[1]](#footnote-2) example of a response from the Registration Request endpoint:

HTTP/1.1 200 OK

Content-Type: application/json

Cache-Control: no-store

Pragma: no-cache

[{"header": {"upv": {"major": 1,"minor": 0},

"op": "Reg",

"appID": "https://uaf-test-1.noknoktest.com:8443/SampleApp/uaf/facets",

"serverData": "Ijycj…VbOnvg", "exts":[{"id":"fido.uaf.uvm", "data":"", "fail\_if\_unknown":true}]},

"challenge": "H9iW9yA9aAXF\_lelQoi\_DhUk514Ad8Tqv0zCnCqKDpo",

"username": "apa",

"policy": {"accepted": [[{

"userVerification": 512,

"keyProtection": 1,

"tcDisplay": 1,

"authenticationAlgorithms": [1],

"assertionSchemes": ["UAFV1TLV"]

}],

…

…],

"disallowed": [{"userVerification": 512,

"keyProtection": 16,

"assertionSchemes": ["UAFV1TLV"]},

…

…]}}]

### Registration Response endpoint

#### Request Syntax

The Client sends the FIDO UAF Registration Response Message (as per section 3.4.4 of the FIDO UAF Protocol Specification) to the Registration Response endpoint using HTTP POST.

The following is a non-normative[[2]](#footnote-3) example of a regResponse Request:

POST /regResponse HTTP/1.1

Host: uaf.prod.signin.nhs.uk

Content-Type: application/json

[{"assertions": [{"assertion": "AT7uAgM…cSNmQ",

"assertionScheme": "UAFV1TLV"

}],

"fcParams": "eyJhcH…hcHAifQ",

"header": {"appID": "https://uaf-test-1.noknoktest.com:8443/SampleApp/uaf/facets",

"op": "Reg",

"serverData": "IjycjPZ…WVbOnvg",

"upv": {"major": 1,

"minor": 0}, "exts":[{"id":"fido.uaf.uvm", "data":"", "fail\_if\_unknown":true}]}}]

#### Registration Response Request Validation

The FIDO UAF Registration Response Message is validated and processed as per section 3.4.6.5 of the FIDO UAF Protocol Specification.

#### Registration Response Response

The response from the Registration Response endpoint will be a list of Registration records stored within the NHS login platform.

The following is a non-normative example of a successful response from the Registration Response endpoint:

HTTP/1.1 200 OK

Content-Type: application/json

Cache-Control: no-store

Pragma: no-cache

[{

"attestDataToSign": "Az6…pSDg==",

"attestSignature": "4PZZ…Rw==",

"attestVerifiedStatus": "VALID",

"authenticator": {

"AAID": "107968223",

"deviceId": null,

"KeyID": "ODY4MjIz",

"status": null,

"username": "8f7…2a4"

},

"authenticator\_string": "107968223#ODY4MjIz",

"AuthenticatorVersion": "0.0",

"PublicKey": "MFkw…pSDg==",

"status": "SUCCESS",

"timeStamp": "1547727594654",

"username": "8f7…72a4"

}]

## FIDO UAF authentication flow

Figure 5 below is a non-normative illustration of the FIDO UAF Authentication flow (taken from the FIDO UAF Architectural Overview document [15]). The process is initiated after FIDO UAF registration has been completed.

The client initiates authentication by invoking the authRequest endpoint of the NHS login platform (1). The NHS login platform generates a FIDO UAF Authentication Request Message and returns this to the client, which passes it to the FIDO Client on the device for processing (2). The FIDO Client interacts with the FIDO Authenticators on the user’s device (3) and creates a FIDO UAF Authentication Response Message corresponding to the original FIDO UAF Registration Request Message. The FIDO Client passes this message back to the client which sends it to the NHS login service as a base64-URL encoded parameter in an OpenID Connect Authentication Request(4) (see fido\_auth\_response parameter in section 3.4.1 of this document). The NHS login platform validates the FIDO UAF Authentication Response Message using the public key for the user stored in the NHS login FIDO registry (5) and if validated treats this as a successful authentication using an authentication vector of “Cm” (see section [5.1.2](#_Authentication_Credentials) of this document).

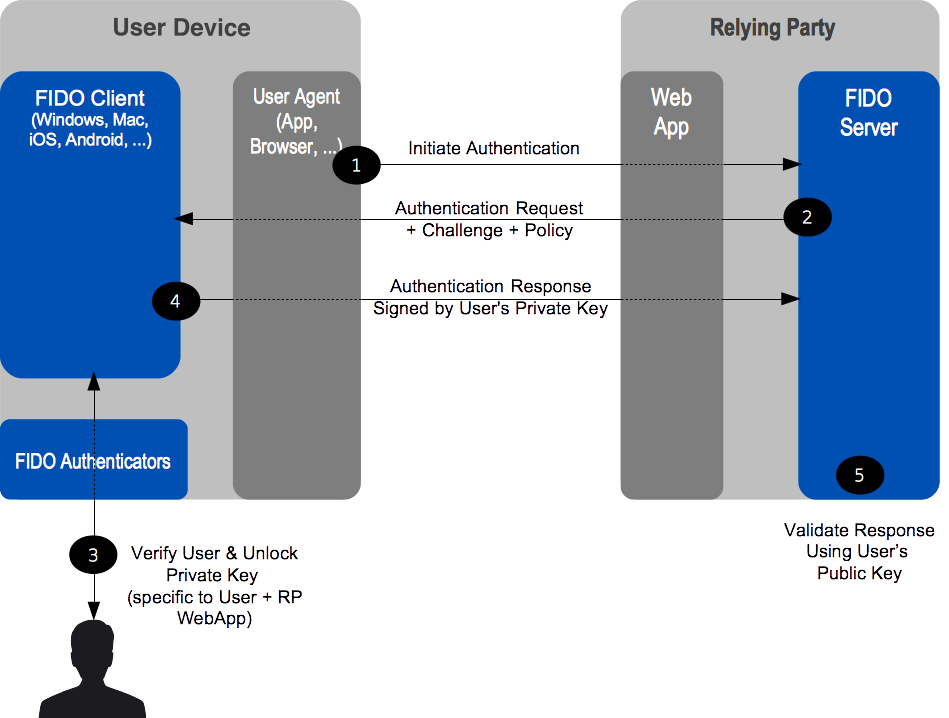


Figure 5 - FIDO UAF Authentication flow

### Authentication Request endpoint

#### Request Syntax

The Client sends the request for a FIDO UAF Authentication Request Message using HTTP GET.

The following is a non-normative example of an authRequest Request:

GET /authRequest HTTP/1.1

Host: uaf.prod.signin.nhs.uk

#### Authentication Request Validation

No specific validation of the request is performed

#### Authentication Request Response

The response from the Authentication Request endpoint will be a FIDO UAF Authentication Request Message as per section 3.5.2 of the FIDO UAF Protocol Specification.

The following is a non-normative[[3]](#footnote-4) example of a response from the Authentication Request endpoint:

HTTP/1.1 200 OK

Content-Type: application/json

Cache-Control: no-store

Pragma: no-cache

[{"header": {"upv": {"major": 1,"minor": 0},

"op": "Reg",

"appID": "https://uaf-test-1.noknoktest.com:8443/SampleApp/uaf/facets",

"serverData": "Ijycj…VbOnvg", "exts":[{"id":"fido.uaf.uvm", "data":"", "fail\_if\_unknown":true}]},

"challenge": "H9iW9yA9aAXF\_lelQoi\_DhUk514Ad8Tqv0zCnCqKDpo",

"username": "apa",

"policy": {"accepted": [[{

"userVerification": 512,

"keyProtection": 1,

"tcDisplay": 1,

"authenticationAlgorithms": [1],

"assertionSchemes": ["UAFV1TLV"]

}],

…

…],

"disallowed": [{"userVerification": 512,

"keyProtection": 16,

"assertionSchemes": ["UAFV1TLV"]},

…

…]}}]

## FIDO UAF deregistration flow

Figure 6 below is a non-normative illustration of the FIDO UAF Deregistration flow (taken from the FIDO UAF Architectural Overview document [15]). The process can be initiated after FIDO UAF registration has been completed.

The client initiates authenticator deregistration by invoking the deregRequest endpoint of the NHS login platform (1). The request MUST include the Access Token obtained from an OpenID Connect Authentication Request, which MUST be sent as a Bearer Token using the Authorization header field, per Section 2 of OAuth 2.0 Bearer Token Usage. The request can either be a list of authenticators to be deregistered (sent using HTTP POST) or a request to remove all authenticators for the user (using a HTTP GET). The NHS login platform generates a FIDO UAF Deregistration Request Message, removes the relevant authenticator records from the NHS login FIDO registry and returns the message to the client. The client passes the FIDO UAF Deregistration Request Message to the FIDO Client on the device for processing (3) and removal of authenticators from the device.

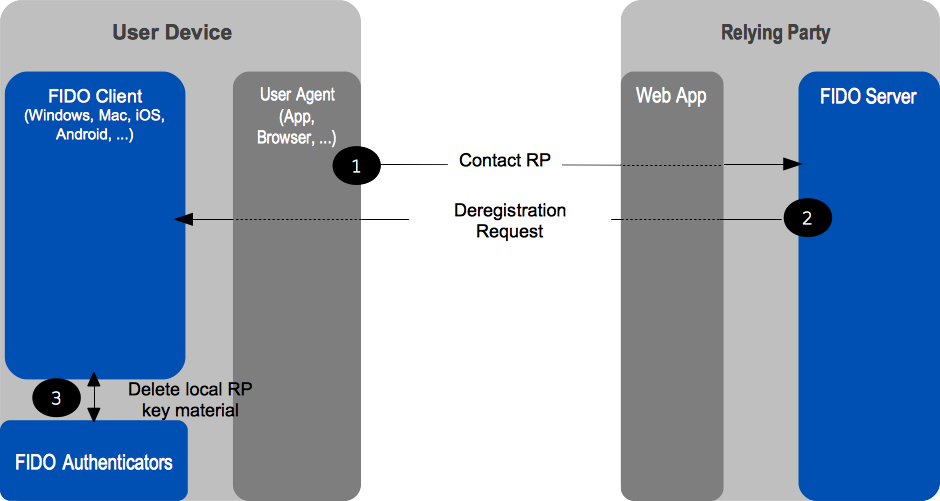


Figure 6 - FIDO UAF Deregistration flow

### Deregistration Request endpoint

#### Request Syntax

The Client either sends a FIDO UAF Deregistration Request Message (as per section 3.6.1 of the FIDO UAF Protocol Specification) to the Deregistration Request endpoint using HTTP POST **or** can make a HTTP GET to the endpoint with no body.

The Access Token obtained from an OpenID Connect Authentication Request MUST be sent as a Bearer Token using the Authorization header field, per Section 2 of OAuth 2.0 Bearer Token Usage.

The following is a non-normative[[4]](#footnote-5) example of a deregRequest HTTP POST Request:

POST /deregRequest HTTP/1.1

Host: uaf.prod.signin.nhs.uk

Content-Type: application/json

Authorization: Bearer SlAV3…2hkKG

[{"header": {"appID": "https://uaf-test-1.noknoktest.com:8443/SampleApp/uaf/facets",

"op": "Reg",

"serverData": "IjycjPZ…WVbOnvg",

"upv": {"major": 1,

"minor": 0}, "exts":[{"id":"fido.uaf.uvm", "data":"", "fail\_if\_unknown":true}]}, "authenticators": [{"aaid": "ABCD#ABCD", "keyID": "ZMC…JfNg"}]}]

The following is a non-normative example of a deregRequest HTTP GET Request:

GET /deregRequest HTTP/1.1

Host: uaf.prod.signin.nhs.uk

Authorization: Bearer SlAV3…2hkKG

#### Deregistration Request Request Validation

The access token is validated, and the NHS login user account details are retrieved.

For HTTP POST processing, each authenticator provided in the message is removed from the NHS login FIDO registry once it has been validated as relating to the NHS login user account.

For HTTP GET processing, each authenticator stored in the NHS login FIDO registry is identified for the NHS login user account and subsequently removed.

The FIDO UAF Registration Response Message is validated and processed as per section 3.4.6.5 of the FIDO UAF Protocol Specification.

#### Deregistration Request Response

The response from the Deregistration Request endpoint will be a FIDO UAF Deregistration Request Message listing the authenticator records removed from the NHS login FIDO registry.

The following is a non-normative example of a successful response from the Deregistration Request endpoint:

HTTP/1.1 200 OK

Content-Type: application/json

Cache-Control: no-store

Pragma: no-cache

[{"header": {"appID": "https://uaf-test-1.noknoktest.com:8443/SampleApp/uaf/facets",

"op": "Reg",

"serverData": "IjycjPZ…WVbOnvg",

"upv": {"major": 1, "minor": 0}},

"authenticators": [{"aaid": "ABCD#ABCD", "keyID": "ZMC…JfNg"}]}]

## Platform Restrictions

This section details specific restrictions implemented within the NHS login FIDO processing.

### Authenticator Policy Matching Criteria

As the NHS login platform is designed for public use, restricting authenticators by device or manufacturer (i.e. via facets) is not an appropriate authenticator matching policy. Instead the platform specifies matching criteria based on User Verification Method, Authentication Algorithm, Assertion Scheme (fixed value of “UAFV1TLV”) and Key Protection.

**User Verification Methods**

The platform will accept authenticators which support Presence AND Fingerprint (1027), OR Presence AND Faceprint (1041), OR Presence AND Handprint (1281) (see FIDO UAF Registry of Predefined Values [16], section 3.1)

If a FIDO UAF Registration Request Message or FIDO UAF Authentication Request Message header contains an extension with an id of “fido.uaf.uvm” then the User Verification Method utilised by the authenticator MUST be returned in the assertion, as a TAG\_EXTENSION tag to the TAG\_UAFV1\_KRD tag (for Registration – see FIDO UAF Authenticator Commands [17] section 6.1.1.1) or TAG\_UAFV1\_SIGNED\_DATA tag (for Authentication – see FIDO UAF Authenticator Commands [17] section 6.1.1.2). The data held in the TAG\_EXTENSION\_ID tag of the extension tag MUST equal “fido.uaf.uvm”, and the data held in the TAG\_EXTENSION\_DATA tag of the extension tag MUST contain the User Verification Method and the value MUST match a User Verification Method value in an accepted Matching Criteria of the relevant Request Message Policy.

**Authentication Algorithms**

The NHS login platform supports the ECDSA family of signing algorithms. Specific values supported are UAF\_ALG\_SIGN\_SECP256R1\_ECDSA\_SHA256\_RAW, UAF\_ALG\_SIGN\_SECP256R1\_ECDSA\_SHA256\_DER, UAF\_ALG\_SIGN\_SECP256K1\_ECDSA\_SHA256\_RAW or UAF\_ALG\_SIGN\_SECP256K1\_ECDSA\_SHA256\_DER. See FIDO UAF Registry of Predefined Values [15] for specific details.

**Key Protection**

The NHS login platform supports on-device key protection mechanisms. Specific values supported are KEY\_PROTECTION\_SOFTWARE, KEY\_PROTECTION\_HARDWARE, KEY\_PROTECTION\_TEE or KEY\_PROTECTION\_SECURE\_ELEMENT. See FIDO UAF Registry of Predefined Values [15] for specific details.

### Public Key Representation Formats

Clients MUST utilise either UAF\_ALG\_KEY\_ECC\_X962\_RAW or UAF\_ALG\_KEY\_ECC\_X962\_DER formats for the TAG\_PUB\_KEY to align with supported Authentication Algorithms.

# Partner Services and Security

## Partner Service Registration

Currently, Partner Services will be registered and onboarded into the NHS login Platform using a standard offline process.

The following information is required is required for the offline registration process:

Table 20: Client Metadata – supplied by client

| Attribute | Description |
| --- | --- |
| Redirect URIs | List of Redirection URI values used by the Client.  The redirect\_uri parameter value used in each Authorization Request MUST exactly match one of these registered Redirection URI values |
| Client Name | Name of the Client to be presented to the End-User |
| Public Key | A copy of the client’s public key is supplied. The public key must be in PEM format and represent a 2048-bit RSA keypair.[[5]](#footnote-6) |
| Scopes | Required scopes |

On successful registration, the following information will be provided

Table 21: Client Metadata – allocated by NHS login

| Attribute | Req? | Type | Description |
| --- | --- | --- | --- |
| client\_id | Mand | String | Id for the client, allocated by the NHS login service |

## Partner Service Authentication

Reference:

* OpenID Connect Core Specification [1], section 9

### Confidential Clients

Partner Services which are confidential clients will use the following authentication mechanism:

* private\_key\_jwt, as per https://tools.ietf.org/html/rfc7523

The authentication mechanism of “none” is not supported for confidential clients

## Client Redirects

The NHS login Platform will only allow redirect URIs which have been pre-agreed during the onboarding process.

Wildcard URIs are not permitted.

HTTP URIs are NOT permitted

URIs with querystring parameters are not permitted[[6]](#footnote-7)

# References

|  |  |
| --- | --- |
| [1] | OpenID Foundation, “OpenID Connect Core 1.0 incorporating errata set 1,” [Online]. Available: https://openid.net/specs/openid-connect-core-1\_0.html. |
| [2] | OpenID Connect Foundation, “International Government Assurance Profile (iGov) for OpenID Connect 1.0,” [Online]. Available: https://openid.bitbucket.io/iGov/openid-igov-profile-id1.html. |
| [3] | Internet Engineering Task Force (IETF), “RFC8485:Vectors of Trust,” [Online]. Available: https://tools.ietf.org/html/rfc8485. |
| [4] | Internet Engineering Task Force (IETF), “RFC7519: JSON Web Token (JWT),” [Online]. Available: https://tools.ietf.org/html/rfc7519. |
| [5] | OpenID Foundation, “OAuth 2.0 Multiple Response Type Encoding Practices,” [Online]. Available: http://openid.net/specs/oauth-v2-multiple-response-types-1\_0.html. |
| [6] | Internet Engineering Task Force (IETF), “RFC6749 - The OAuth 2.0 Authorization Framework,” [Online]. Available: https://tools.ietf.org/html/rfc6749. |
| [7] | Internet Engineering Task Force (IETF), “RFC6750: OAuth 2.0 Bearer Token Usage,” [Online]. Available: https://tools.ietf.org/html/rfc6750. |
| [8] | Internet Engineering Task Force (IETF), “RFC7515 - JSON Web Signature (JWS),” [Online]. Available: https://tools.ietf.org/html/rfc7515. |
| [9] | Internet Engineering Task Force (IETF), “RFC3339 - Date and Time on the Internet: Timestamps,” [Online]. Available: https://www.ietf.org/rfc/rfc3339.txt. |
| [10] | OpenID Foundation, “OpenID Provider Authentication Policy Extension 1.0,” [Online]. Available: http://openid.net/specs/openid-provider-authentication-policy-extension-1\_0.html. |
| [11] | NHS Digital, “Access Tokens and Audit (JWT),” [Online]. Available: https://developer.nhs.uk/apis/spine-core/security\_jwt.html. |
| [12] | NHS Digital, “DCB3051 Identity Verification and Authentication Standard for Digital Health and Care Services,” [Online]. Available: http://digital.nhs.uk/isce/publication/dcb3051. |
| [13] | Cabinet Office, “GPG44 - Authentication credentials for online government services,” [Online]. Available: https://www.gov.uk/government/publications/authentication-credentials-for-online-government-services. |
| [14] | Cabinet Office, “GPG45 - Identity proofing and verification of an individual,” [Online]. Available: https://www.gov.uk/government/publications/identity-proofing-and-verification-of-an-individual. |
| [15] | FIDO Alliance, “FIDO Universal Authentication Framework,” [Online]. Available: https://fidoalliance.org/specifications/download/. |
| [16] | FIDO Alliance, “FIDO UAF Registry of Predefined Values,” [Online]. Available: https://fidoalliance.org/specs/fido-uaf-v1.0-ps-20141208/fido-uaf-reg-v1.0-ps-20141208.html. |
| [17] | FIDO Alliance, “FIDO UAF Authenticator Commands,” [Online]. Available: https://fidoalliance.org/specs/fido-uaf-v1.0-ps-20141208/fido-uaf-authnr-cmds-v1.0-ps-20141208.html. |
| [18] | OpenID Foundation, “Enhancing OAuth Security for Mobile Applications with PKCE,” [Online]. Available: http://openid.net/2015/05/26/enhancing-oauth-security-for-mobile-applications-with-pkse/. |
| [19] | NHS Digital, “DCB3051 Identity Verification and Authentication Standard for Digital Health and Care Services,” [Online]. Available: http://digital.nhs.uk/isce/publication/dcb3051. |

1. The contents have been truncated for legibility [↑](#footnote-ref-2)
2. The contents have been truncated for legibility [↑](#footnote-ref-3)
3. The contents have been truncated for legibility [↑](#footnote-ref-4)
4. The contents have been truncated for legibility [↑](#footnote-ref-5)
5. An example use of OpenSSL to generate a RSA public/private key pair and then extract the public key is as follows:

   openssl genpkey -algorithm RSA -out ../keys/private\_key.pem -outform PEM -aes-256-cbc -pass file:../keys/password.txt -pkeyopt rsa\_keygen\_bits:2048

   openssl rsa -pubout -in ../keys/private\_key.pem -out ../keys/public\_key.pem -passin file:../keys/password.txt [↑](#footnote-ref-6)
6. The 'state' parameter described in section 3.4 is designed to support conveying dynamic request content within the OIDC authentication request. [↑](#footnote-ref-7)