THE Berlin GROUP
A EUROPEAN STANDARDS INITIATIVE

6-weeks market consultation
NextGenPSD2 Access to Account Interoperability Framework

COMMENTS SHEET V1.00

Distribution: Publicly available

How to use this document:

1. Please fill in your professional details in the section below
2. For each line in the comments sheet, please assign a sequence comment N° (in ascending order)
3. For each line in the comments sheet, please assign a comment ID:
   - T = Technical comment (correction or clarification of a concrete technical requirement defined by the specification standards, not altering or expanding any functional features). Please provide an exact reference of the comment to the location in the applicable document.
   - E = Editorial comment (correction or clarification of a topic description without implying any technical changes. Only the descriptive part of the specification might be affected). Please provide an exact reference of the comment to the location in the applicable document.
   - G = General comment (any other type of comment, which may also include altered or expanded functionality for which a justification must be provided)

Fields of the form which are marked grey must not be completed by the contributor.
Please send your completed comments sheet exclusively by email to: consultation@berlin-group.org until Friday 17 November 2017 (COB).

Date:	14-11-17
Name of contributor:	Torsten Lodderstedt
Email of contributor:	torsten@lodderstedt.net
Telephone of contributor:	+4915116713400
Title/Position:	BG Liaison
Organisation:	OpenID Foundation Financial API Working Group
Type of Organisation (FinTech/ASPSP/both):	Both
Address:	2400 Camino Ramon, Suite 375, San Ramon, CA 94583
Country:	United States
Your reference:	

To ensure open and transparent consultation, the Berlin Group may wish to publish the received market feedback on the Berlin Group public website, only mentioning the organisation name and their submitted comments.

THE Berlin GROUP
A EUROPEAN STANDARDS INITIATIVE

6-weeks market consultation
NextGenPSD2 Access to Account Interoperability Framework

COMMENTS SHEET V1.00

Distribution: Publicly available

Document:

01. NextGenPSD2 Access to Account Interoperability Framework - General Introduction Paper V099_20171002.pdf

Comment (N° / ID)	Comment/question (when applicable with justification/rationale or reference section/page n°)	Suggested Resolution (alternative)	Agreed Resolution (Workgroup)

THE Berlin GROUP
A EUROPEAN STANDARDS INITIATIVE

6-weeks market consultation
NextGenPSD2 Access to Account Interoperability Framework

COMMENTS SHEET V1.00

Distribution: Publicly available

Document:

02. NextGenPSD2 Access to Account Interoperability Framework - Operational Rules V099_20171002.pdf

Comment (N° / ID)	Comment/question (when applicable with justification/rationale or reference section/page n°)	Suggested Resolution (alternative)	Agreed Resolution (Workgroup)

THE *Berlin* GROUP
A EUROPEAN STANDARDS INITIATIVE

6-weeks market consultation
NextGenPSD2 Access to Account Interoperability Framework

COMMENTS SHEET V1.00

Distribution: Publicly available

Document:

03. NextGenPSD2 Access to Account Interoperability Framework – Implementation Guidelines V099_20171002

Comment (N° / ID)	Comment/question (when applicable with justification/rationale or reference section/page n°)	Suggested Resolution (alternative)	Agreed Resolution (Workgroup)
1/E	All the sub-clauses and figures should be assigned a unique number so that they can be addressed properly.	Add numbers, e.g., according to ISO/IEC Directive Part 2.	
2/G	The security protocol described here does not seem to have the rigor of OIDF Financial API (FAPI) Security Profiles (Part 1 and Part 2.), which is adopted by Open Banking (UK), recommended by the Japanese Banker's Association, and under the consideration by the FS-ISAC (USA). It should be extended to cover the same or simply refer them.	Instead of coming up with a new security profile, we recommend to refer to FAPI Part 1 and Part 2. . Alternatively, the protocol must undergo a rigorous security threat analysis.	
3/G	For the functional API specification, it is much better to utilize Open API (a.k.a Swagger) specification.	Adopt Open API specification.	
4/E	TAN is not defined.	Please define.	
5/G	Details of the TPP authentication against XS2A is not given and thus it does not seem to be interoperable specification. This is important, as the TAN must be tied to the TPP authentication session to be secure.	Please specify the details.	
6/G	It unclear how onboarding is supposed to work, i.e. via Client Registration. This is still possible even if the ASPSP can impose no contact on the TPP.	Please describe the way onboarding shall work. Perhaps it could refer to dynamic client registration specification.	
7/G	Using a certificate identifying a company to identify it's software in every operation is not best practice. It would mean that all services that a TPP / ASPSP operates would need access to a single private key. This would increase the chance of that key being compromised and make key rotation harder..	The UK Open Banking Implementation Entity is working on a solution that takes advantage of dynamic client registration with software statements as defined in RFC7591 - the OAuth 2.0 Dynamic Client Registration Protocol. This standard, which builds on	

	There is an important difference between a company and the software it produces: Many registered TPPs will have multiple software products A software product will often have a shorter lifetime than a company A software product may require a subset of the permissions that a company is granted Our strong recommendation, based on significant PKI experience, is that the eIDAS cert with the role is used for onboarding a TPP with an ASPSP, but not for subsequent interactions. Ensuring that a TPP hasn't had its permissions revoked and therefore its eIDAS certificate revoked can be achieved through other means, e.g. an intermediate CA that syncs with all eIDAS CAs and issues the certificates used by the software or a backchannel process that syncs revocation according to a schedule that matches the agreed liability regime. While it may seem simpler for each ASPSP to perform an OCSP check on an eIDAS certificate issued to the company on every interaction there are practical difficulties with this: The TLS endpoints at the ASPSP would need to trust and make outbound requests to every eIDAS CA (QTSP) There is likely to be an increased latency and a chance for requests to fail	OAuth 2.0 can help lower the barrier to entry for TPPs while at the same time separating the identification of the TPP and it's software. Consider referring to it.	
8/T	3. Transport Layer Just stating TLS 1.2 is not adequate. Cipher suite needs to be specified as well because "null" cipher etc. does not serve the purpose.	The recommendations for Secure Use of Transport Layer Security in BCP195 shall be followed, with the following additional requirements: TLS version 1.2 or later shall be used for all communications.	

		A TLS server certificate check shall be performed, as per RFC6125. Probably, aligning to FAPI Part 1 and 2 would be a good idea.	
9/G	4.2 [signHTTP] is not a standard nor it is on the way to become standard. It has not and is not going under the rigor of the standardization process. It is just an individual IETF draft that any person can come up at any day.	Use RFC7515 JWS instead.	
10/G	4.2 [signHTTP] does not seem to address the integrity protection for an HTTP redirect, which happens to be the most critical one. Direct communication between the client and the server is TLS protected and the signature is not critical as long as TLS is properly used.	Use RFC 7515 JWS instead as in OAuth JWS Authorization Request.	
11/E	4.3 Optional Usage of OAuth2 as a Pre-Step OAuth2 should be indicated by which RFC it means. There are many OAuth 2 related specifications. It probably means RFC6749.	Change OAuth2 to OAuth 2.0 [RFC6749].	
12/G	4.3 Optional Usage of OAuth2 as a Pre-Step It does not make sense to use RFC6749 this way. RFC6749 should be used to obtain the PSU's authorization rather than as a pre-step to obtain something akin to PSU's password. Also, it is a much better practice to use the client specific tokens instead of PSU password. So, this should be a recommended way instead of an optional way.	Adopt OAuth 2.0 framework and profile it to suit the needs as the default mechanism to obtain the PSU's consent as well as for the protection of the XS2A API.	
13/G	4.3 Consideration should be given if the bearer token is adequate for some of the operations. OIDF FAPI WG maintains that it is not and recommending the use of sender constrained tokens in order to prevent access token leakage.	Use of [MTLS] or {TOKB] should be considered for sending money, etc. [MTLS] https://datatracker.ietf.org/doc/draft-ietf-oauth-mtls/	

		[TOKB] https://datatracker.ietf.org/doc/draft-ietf-oauth-token-binding/ Please consult https://tools.ietf.org/html/draft-ietf-oauth-security-topics for a discussion of the attack angle	
14/G	4.3 With "user password grant", I trust it to mean Resource Owner Password Credentials. As RFC6749 points out in section 10.7, the authorization server and client SHOULD minimize use of this grant type and utilize other grant types whenever possible.	Use redirection flow with In-App web browser instead.	
15/G	4.3 The use of "user password grant" makes the liability separation unclear.	Mention this liability unclearness in the text.	
16/E	4.3 There is no "Authorization Flow" in RFC6749.	If it means "Authorization Code Grant", then replace with it.	
17/G	4.6 The use of the hyperlink is one step forward towards HATEOAS and nice but it also increases the attack surface compared to the set of pre-determined static URLs which can be evaluated at the registration time statically.	Please specify how the TPP can learn that the response is coming back from the authoritative source. Otherwise, this idea of following the links seems to be dangerous.	
18/E	P.53 "redirect" value in "_links" does not have the schema. It probably is better to have one.	Add https://.	
19/E	5.1 Redirect SCA approach, the first figure. On the surface, it appears as if the user agent such as a web browser can be used as a PSU client, but in the subsequent description, it becomes apparent that it is not as the redirection message still uses HTTP response code 200. This is confusing to the readers.	Add a note in the figure that the client has to be able to interpret the "redirect" value in the "_links" payload in the JSON to perform the redirect.	
20/G	5.1 Embedded SCA Approach without SCA method (e.g. Creditor in Exemption List) is essentially a phishing and should be banned for the sake of the general internet health.	Remove it or make it optional for the ASPSPs to implement.	

	Also, this approach severely limit the possibility for the ASPSP to evaluate the risk associated with the PSU authentication. It has to rely on the TPP for it but the liability resides in ASPSP under the PSD2 regime. This is not a good situation for ASPSPs. See Appendix A for the comparison of the Redirect/Embedded/Decoupled approach.		
21/G	5.1 Embedded SCA Approach with only one SCA method available The protocols performing SCA in the embedded mode is incompatible with the way any modern phishing resistant authenticator such as W3C Web Authentication works. This is because such authentication system mutually authenticate the parties and detects any man-in-the-middle. Thus, it is actually imposing the restrictions on the ASPSP's ability to innovate.	Add a note that this method may break at anytime once the ASPSP upgrades their authentication method to protect PSU. Make the approach optional.	
22/G	5.1 Embedded SCA Approach It is not clear if the authorization given in this context constitutes a PSU's explicit consent. There is no way for the ASPSP to know if the privacy notice and TOS were properly displayed to the PSU and PSU made an action to accept it. (TAN requires an action by the PSU but whether the notices were properly displayed cannot be known by the ASPSP so it may be meaningless.) Since the act actually constitute the transfer of personal data to a third party, this is quite important in the term of GDPR. Unless there is a legal provision that ASPSP is exempted from the GDPR provisions, this approach seems to be too dangerous.	Make the approach entirely optional for ASPSPs.	
23/G	6.1.1 Decoupled SCA Approach The figure apparently talks about "access token". Is this an OAuth access token? It should be clarified.	The CIBA spec should be considered as it supports the ASPSP notifying the TPP when auth has taken place - rather than the TPP continually pinging the ASPSP.	

	Also, there is a specification that is being developed called "CIBA"[1]. It probably should be mentioned as well. [1] http://openid.net/specs/openid-connect-modrna-client-initiated-backchannel-authentication-1_0.html	CIBA also supports tokens - and therefore long-lived consent with refresh tokens.	
24/G	8 Combination of AIS and PIS Services "In a context, where the consent management for account access is fully provided by the Oauth2 model, the corresponding access tokens will support this feature analogously." This is a key point - OAuth 2.0 [RFC6749] allows a clean separation between a PSU authorising access and the presentation of a token representing that authorisation decision for ongoing API access. If OAuth 2 is seen as a positive for more complex services, then surely there is a case for using it throughout the API.	If pass through methods (SCA embedded mode) must be supported, then they should be separated out from the rest of the API. The direction of travel in the industry suggests that such methods will be deprecated in coming years. For example no matter what PSD2 says, if more banks start experiencing these type of attaks: https://www.theregister.co.uk/2007/04/19/phishing_evades_two-factor_authentication/ then banks will move to the use of phishing resistant credentials, which it will be impossible to use in a "pass-through" API.	
25/G	How shall authentication work in corporate scenarios? One would assume such scenarios require different authentication and authorization mechanisms to be support by the BG API. The current design would require to add another (or a couple of) mode(s) and would all interested TPP implementers to change their implementations accordingly.	Separate authentication & authorization from the actual API by utilizing the OAuth 2.0 framework and its abstractions.	
26/G	We are really concerned about the embedded mode for the following reasons: We don't think the flow can be made really secure if the PSU credentials flow through systems of the different parties. All the typical processes involved in securing systems, such as security design, (source code) audits, pen-testing, monitoring & abuse detection are spread across those systems/entities. Moreover,	Clearly define the liabilities of each of the participating parties.	

	due to the late binding between TPP and bank, it's unclear how both parties cooperate in achieving the protection goal. It's unclear to me what party is liable for proper credential handling and potential security incidents caused by credential exposure/leakage to adversaries? Please specify precisely.		
27/G	As far as we understand, Berlin Groups wants to support a usage pattern where a TPP offers PSD2 services to other parties (e.g. merchants), who are not registered TPPs. In such cases, it would be important to include information about the final destination of the data or transfer into the consent process. For PISPs, this data could be channeled through the remittance information, but there is no such a concept in the AIS.	add such information to the consent object/process in order to help PSUs to meet informed decisions.	
28/G	Explicit consent handling by the ASPSP in embedded mode is bound to SCA. This means the ASPSP cannot ask the PSU for consent in cases without SCA, e.g. ● Payment initiation service requests that are subject to an exemption ● Account information service requests on whitelisted accounts As a consequence, there is either no PSU consent in those cases or it is gathered by the TPP.	ASPSP would be in a much better position wrt GDPR if they would always gather the PSU's consent. We recommend to enhance this mode to allow for explicit consent in all cases	
29/G	The spec excludes consent handling for PIIS and suggests out-of-band consent handling through the ASPSP's online banking portal. That leaves open the following questions: ● What will the ASPSP tell the PSU? Could it ask the PSU to unlock the PIISP access with her Bank?	We recommend you to extend the spec to cover those topics or to include PIISP consent handling into the spec.	

	• Will the PIISP be able to assist to find the right place? Is it "the Online-Banking" or is it more specific, Please go to "mybank.de/piisp_authz"? • Will the PIISP be able to tell the PSU what to do there? Is this process supposed to work the same with every ASPSP? • How is the PSU supposed to identify the PIISP towards the ASPSP? Using a name, a URL, a DN from the PIISPs X.509 certificate? • How is the ASPSP supposed to check the PIISPs authorization given there will be no direct interaction, which is the prerequisite to check the PIISPs certificate.		
30/G	The Redirect mode is underspecified. For example, there is no text, how the PSU's user agent is redirected back to the TPP and how the respective redirect URL is determined by the ASPSP. Moreover, there is no text of how typical security threats on redirect-based protocols (e.g. open redirection, CSRF, injection) are mitigated.	We recommend to build the redirect mode based on OAuth 2.0 grant type "Authorization Code", as it has all this mechanics built-in.	
31/G	OAuth is mentioned as an alternative for AISP consent handling but not specified (e.g. scopes).	Please add an explanation.	
32/G	This spec specifies and relies on a proprietary framework for authentication, authorization and consent handling. This poses the following challenges:	We recommend you to consequently adopt OAuth 2.0, more specifically OIDF Financial API (FAPI) Security Profiles (Part 1 and Part 2.). OAuth 2.0 is the de facto API access authorization standard for RESTful/HTTP-based	

	• This framework and the API need to go through a systematic and thorough review process in order to ensure its effectiveness against all possible security threats. • Developers have to learn a new authorization framework, increasing the implementation burden for them.	APIs. It's a mature protocol (in practical use since 2010), which went through a systemic treat analysis, a broad security review within IETF and by other communities. As a side-effect, use of OAuth/OpenId Connect in combination with the redirect mode allows the ASPSP to handle subsequent transactions in a session context potentially offering its PSUs a SSO experience (without the need for the TPP to handle this context). Using OAuth, it would also be possible to gather consent for PIISP and to combine user consent for different PSD2 use cases, e.g. PIISP and PISP.	
33/G	5 Payment Initiation Service The current protocol directly initiates the payment in the SCA/consent gathering process. The design therefore does not allow for split, partial, delayed, or recurring payments.	Consider to enhance the design to separate SCA/consent and actual submission of the payment process (in the same way STET and OpenBanking UK do). Dynamic linking between SCA and initiation can be achieved by way of associating the context of the transaction with an access token.	

Appendix A -- Comparison chart of the Authentication Approaches

	Redirect	Embedded	Decoupled
Supports SCA based on one-time-passcodes	Yes - but these are vulnerable to phishing	Yes - but these are vulnerable to phishing	Yes - but these are vulnerable to phishing
Supports SCA elements based on knowledge	Yes - but these are vulnerable to phishing	Yes - but these are vulnerable to phishing	Yes - but these are vulnerable to phishing
Supports phishing resistant SCA elements	Yes	No - phishing is a huge problem and the direction of travel for authentication on the Internet is to phase-out OTPs and move towards phishing resistant authentication.	Yes
Supports W3C Web Authentication standard	Yes	No	Yes
Supports the ASPSP getting explicit consent from the user to release data / funds.	Yes - the ASPSP can get explicit consent from the end user as part of the redirect flow. This is important for GDPR	No - the ASPSP has to rely on the consent the TPP has collected - this may not be compliant with GDPR.	Yes

Supports banks retrieving sophisticated fraud markers	Yes - ASPSPs have significant experience in reducing fraud by collecting data points in the redirect journey	No - the bank receives no fraud markers beyond what is sent by the TPP.	No - because the authentication and consumption devices are decoupled
Requires ability for ASPSP to send out of band notifications to an “app”	No	No	Yes - this could limit adoption
Has established, mature international standards in wide use and known threat models.	Yes - the OAuth 2 family of standards (including OpenID Connect) is in wide use and solves these problems	No. No other industry uses pass-through. OAuth did have a “user:password grant” but this wasn’t intended to be used by third parties and has been deprecated.	Not yet - work is being done in this area by the OpenID Foundation, e.g. the Client Initiated Backchannel Authentication spec.