**Scaling Dynamic Client Registration and Endpoint Discovery for Open APIs - DRAFT**

**February, 2019**

**Business Problem**

The API ecosystem today places a burden on data holders (both providers and payers, in the healthcare sector) to register external apps and differentiate the APIs to which a given app is granted access based on the difference in applicable policy requirements for a given use case.  A burden is also placed on app developers to register their apps with many different data holders, for one or more use cases.  There is a non-negligible cost associated with repeatedly meeting the eligibility criteria of each data holder in order to provide app functionality to their users (patients, providers and payers).

This problem applies to many scenarios including:

* Enabling the sharing of health information held by a Covered Entity with the application of a consumer’s choice without special effort.
* Enabling the sharing of Payer held data with Providers.
* Enabling the sharing of Patient Generated Health Data (PGHD) with Covered Entities
* Enabling the sharing of consumer aggregated data with Benefit determination process
* And many more.

There is a great deal of momentum advancing the consumer access use case. From a standards development and adoption perspective it is important that a mechanism established to support consumer access is extendable to enable provider-to-provider, cross-organizational exchange and other use cases that are expected to emerge.

In anticipating the different scenarios where a trust enabling mechanism may be employed by the community it is important to acknowledge that various kinds of “validation” will be required to provide the utility required by a relying party to make a decision about sharing data and access via APIs that they manage. Therefore, a solution designed to address this general need should be able to support many forms of “validation”. The term “validation” is used here in the broad sense, where a relying party may have one or more local policies (perhaps an extension of regulatory compliance requirements) that drive what conditions must be met to permit access to a given application for a given API. For example, enabling a consumer controlled application to access a relying party’s Consumer facing API’s for the purpose of supporting the consumer’s access to their health information may be facilitated by a relying party’s ability to validate[[1]](#footnote-1) a consumer controlled application’s claim that it has self-attested to a set of industry-established requirements (such as the CARIN Alliance pledge), while other use cases (such as enabling a consumer application to submit data for Benefit Determination purposes) may require one or more “Certifications” or “Accreditations”.

The label we use for the organizations that provide this validation is “Endorser”.[[2]](#footnote-2) In the best of all possible worlds there would be many “Endorsers” that are trusted by relying parties for a given characteristic, relevant to the relying party’s determination of whether to trust a given app for a given scenario, to ensure a competitive market of Endorsers.

As a relying party, it would be ideal to have a mechanism that enabled me to electronically evaluate the validity of claims made by an application registering to have access to the many scenarios supported by my organization’s APIs.

We believe that a standards-based mechanism that enables many “Endorsers” to apply their endorsements to many applications in a trustworthy, verifiable way that is flexible enough to differentiate the validation of application claims (and thereby permit access for different scenarios), such that its adoption would be facilitated, and that would be extendable to support future scenarios in a market driven manner, would be ideal.

In the following sections we describe how the envisioned solution would apply for the “consumer access to their PHI” scenario. We use this use case as an exemplar as it is a well-understood scenario with immediate need across the healthcare eco-system.[[3]](#footnote-3)

**Highest Level User Stories**

As a data holder, I do not want to have to expend resources unnecessarily to vet the apps to which my organization wants to grant access to Protected Health Information.  Under HIPAA my organization is obligated to share health information with consumer-controlled apps selected by the patient. At the same time my organization has an obligation to protect our information assets from bad actors. As there may be 100s of different apps that could be selected by a consumer, my organization would incur significant cost burdens in order to comply with the intent of applicable law. I need a mechanism that allows me to dynamically register an Application that has claims validated by “Endorsing” entities that I choose to trust.

As an app developer, the greatest barrier to scaling the use of my innovative offering is the cost of adoption.  For example, for a consumer-controlled application to be effective, it must be able to access data from all of the providers who have treated a user[[4]](#footnote-4).  If, as a developer, I must go through a separate process with each provider system used by the consumer and in each case demonstrate in a one-off onboarding process the qualifications of my solution, I will never be able to scale my offering beyond a limited set of data holders.

As a policy maker, I want to be able to enable innovation, competition and broad choice of solutions to benefit consumers, providers and payers. In the best of all possible worlds the trust mechanism would be able to support current needs and be future proof for other scenarios.

As a consumer, I want to be able to use the application of my choice and be able to make better care decisions including the ability to pick the best provider for my needs without being forced to switch applications based on the decisions of the payers and providers I happen to be using at a given point in time.

These considerations motivate the construction of a reusable approach to app registration, user, and API endpoint participation that will allow the use of APIs at the scale needed to achieve optimal benefit by all stakeholders. The solutions proposed here extend OAuth 2.0 workflows and Dynamic Client Registration, which are in a position to leverage digital key material but which currently rely on one-off pairwise agreements to do so, to add assurance for and about all parties involved in the API ecosystem.

This is achieved through 1) verification of the “identity” of each party such as by a certification authority and 2) subsequent relying party validation, via the mechanism, of the trustworthiness of claims made by an application about its endorsements 3) by a registered “Endorser” that applies its “endorsements” in a verifiable way.

Being able to validate that a participant holds a key validated by a trusted certification authority eliminates the need for pairwise key exchange and helps dynamically scale the network of trusted participants. The envisioned mechanism foresees each Application being registered as well as each “Endorser”. An Endorser would be able to apply or rescind its endorsement(s) against a registered Application.

This trust model could be extended to require relying parties that chose to adopt the mechanism to publish their API endpoints for discovery by registered consumer applications. This would create a virtuous circle where Application Developers, Endorsers and Relying parties all benefit by participating, through:

1) an ability to validate the trustworthiness of application certifications or endorsements,

2) better scaling of the application, user, and endpoint discovery process

3) the publication of the processes performed by a given Endorser to enable relying parties to implement their access point processes, and

4) other secondary benefits such as providing a registry of available apps for discovery by consumers and policy makers.

**Existing Technical Frameworks**

* The Unified Data Access Profiles (UDAP, <http://www.udap.org>) describe methods that incorporate the use of trusted digital certificates into open API workflows, primarily as extensions to commonly deployed OAuth 2.0 workflows. These profiles have been successfully tested at several HL7 FHIR connectathons. In UDAP Dynamic Client Registration (<http://www.udap.org/udap-dynamic-client-registration.html>), trusted digital certificates are used to help scale the client registration process, and UDAP Certifications and Endorsements (<http://www.udap.org/udap-certifications-and-endorsements.html>) are used to communicate information about an app from trusted Endorsers, also referred to as UDAP “App Certifiers”. Here is a high-level summary:

1. Client application developer applies for a UDAP certificate to use in UDAP Dynamic Client Registration.
2. The UDAP-compatible Certification Authority follows its policies, compliant with the community within which the client app is requesting participation, for vetting and then issuing a certificate according to the applicable certificate profile.
3. Developer implements UDAP Dynamic Client Registration and uses their certificate at UDAP-compatible endpoints to register dynamically.
4. Client Applications can include certifications and endorsements issued by third party App Certifiers in their dynamic registration request, validating attributes associated with the application, the app’s compliance with a given certification program, or the Certifier’s endorsement of the app for a certain use case. These are digitally signed by the App Certifier using a UDAP certificate issued to the App Certifier. A relying party can evaluate the signature on each certification and endorsement to determine if it comes from an App Certifier that it trusts, allowing for a relying party to easily establish its own policies about which certifications and endorsements should be considered valid for a particular use case or class of application.
5. Compliant servers also advertise their implementation of UDAP Dynamic Client Registration, and adhere to policies as set by relevant communities in which they participate. These participants additionally register for and use UDAP certificates.
6. API Endpoints of UDAP implementers may be published in a directory that can be accessed by trusted UDAP client applications.

UDAP Client Application attributes include items such as Company (Client application distributor) Name, Application Healthcare Category (Covered Entity, Business Associate, etc.), App Name and Version, App Redirect URI, Privacy Policy URL (Including detailed information about Data Security Practices and User Identity Assurance), and other useful attributes about the app or its developer.

UDAP Endpoint and User Attributes may include items such as Company (Data Holder) Name, API Endpoint Domain, Organization’s Healthcare Category (Covered Entity, Business Associate, etc.), Data Holder’s API Terms of Use URL, and other useful attributes about a FHIR endpoint or querying endpoint.

* As part of the Blue Button 2.0 API project at CMS the Pre-OAuth Entity Trust (POET) specification was proposed as a means to verify application endorsements and their provenance. POET can be used as an API authentication for the Dynamic Client Registration Protocol (DCRP) endpoint.

The POET specification was developed into a reference implementation under the auspices of the non-profit TransparentHealth organization. The original specification and the associated reference implementation are available here:

* <https://github.com/TransparentHealth/poet>
* <https://github.com/transparenthealth/python-poetri>

POET uses a JWT signed with the private key from an endorsing body. The DCRP process for an OAuth server can then use the Endorsing Body’s public key to verify the signature. A JWT might look something like:

{"manifests": [{"https://apps-dstu2.smarthealthit.org/cardiac-risk":   
 "client\_name" : "Cardiac Risk App",   
 "client\_uri": "https://apps-dstu2.smarthealthit.org/cardiac-risk/",   
 "logo\_uri" : "https://gallery.smarthealthit.org/img/apps/66.png",  
 "software\_id" : "cadiac-risk-app",   
 "redirect\_uri" : ["https://apps-dstu2.smarthealthit.org/cardiac-risk/redirect"],  
 "poet\_endorsements": ["JWT1", "JWT2"]}, {"https://apps-dstu2.smarthealthit.org/bp-centiles":   
 "client\_name" : "Blood Pressure App",   
 "client\_uri": "https://apps-dstu2.smarthealthit.org/bp-centiles",   
 "logo\_uri" : "https://gallery.smarthealthit.org/img/apps/20.png",  
 "software\_id" : "blood-pressure-app",   
 "redirect\_uri" : ["https://apps-dstu2.smarthealthit.org/bp-centiles/redirect" "poet\_endorsements": ["JWT3", "JWT4"]} ]   
}

POET has been presented to experts at the Internet Identity Workshop where the consensus is that it adds an important capability to the OAuth2.0 specification by enabling data holders implementing OAuth2.0 with POET support to enable DCRP for developers presenting a verifiable JWT.

**Next Steps**

At the FHIR Connectathon20, in San Antonio, TX on January 12-13 we had a number of informal meetings with community members who encouraged our continued work in this area. We believe that the timing is right to establish a FHIR Connectathon21 Track specifically related to this effort with broader participation from the community.

As a first step we are sharing this document as a way to solicit input from the FHIR community in order to refine the content of this concept paper and to identify parties interested in participating in a workstream specific to this activity under the auspices of the FHIR community.

**Authors**

Aaron Seib, NewWave

Julie Maas, EMR Direct

Mark Scrimshire, NewWave

Alan Viars, Videntity/Transparent Health

1. There is nothing about this approach that precludes relying parties from implementing alternative paths to recognizing app’s that they approve. Further there is nothing to prevent app developers from working directly with data holders to get access to APIs granted by the governance process of the local organization. The objective of the proposed mechanism would be to help reduce burdens related to the process where ever possible recognizing that alternative approaches at the local level are critical as well. [↑](#footnote-ref-1)
2. An unfortunate label and one that we’d welcome a better alternative for, as “Endorser” carries connotations that the endorsement goes beyond the scope of the “Endorser’s” processes and implies that the app is recognized for more than the specific scope of the “Endorser’s” endorsement. Clearly, endorsing an app for “user friendliness” has no merit in evaluating its clinical utility or any other attribute for that matter. [↑](#footnote-ref-2)
3. Our focus is on healthcare but it is important to recognize that the need for this solution is generalizable to other industries as this problem exists across the digital universe. [↑](#footnote-ref-3)
4. Similar to the term “Endorser” the label “user” is not always satisfactory based on the use case/scenario in context. Often referred to as Patients or Members or Clients – a user of a consumer-controlled application may have different labels. [↑](#footnote-ref-4)