Integrating the Healthcare Enterprise

IHE ITI
Technical Framework Supplement

Internet User Authorization (IUA)

Draft for Public Comment

Date: June 03, 2013

Author: ITI Technical Committee

Email: iti@ihe.net

Foreword

This is a supplement to the IHE IT Infrastructure Technical Framework V9.0. Each supplement undergoes a process of public comment and trial implementation before being incorporated into the volumes of the Technical Frameworks.

This supplement is published on June 03, 2013 for Public Comment. Comments are invited and may be submitted at http://www.ihe.net/iti/iticomments.cfm. In order to be considered in development of the Trial Implementation version of the supplement, comments must be received by July 03, 2013.

This supplement describes changes to the existing technical framework documents.

"Boxed" instructions like the sample below indicate to the Volume Editor how to integrate the relevant section(s) into the relevant Technical Framework volume.

Amend section X.X by the following:

Where the amendment adds text, make the added text **bold underline**. Where the amendment removes text, make the removed text ~~**bold strikethrough**~~. When entire new sections are added, introduce with editor's instructions to "add new text" or similar, which for readability are not bolded or underlined.

General information about IHE can be found at: www.ihe.net.

Information about the IHE IT Infrastructure domain can be found at: http://www.ihe.net/Domains/index.cfm.

Information about the organization of IHE Technical Frameworks and Supplements and the process used to create them can be found at: http://www.ihe.net/About/process.cfm and http://www.ihe.net/profiles/index.cfm.

The current version of the IHE IT Infrastructure Technical Framework can be found at: http://www.ihe.net/Technical_Framework/index.cfm.

CONTENTS

Introduction to this Supplement

Problem Statement

This profile is motivated by customer requirements for authorizing network transactions, when using HTTP RESTful transports. Authorization means that the user, patient or provider, has legitimate access to this RESTful service. The authorization includes identifying the user, device, and or application that is making the request to the RESTful server, so that server can make further access control decisions. The RESTful services may include user driven browser activity, downloaded applications, and automatic devices. The existing IHE ITI XUA profile fills these needs for the SOAP transport based transactions. The existing IHE ITI EUA profile fills these needs for various different transports within a single enterprise environment, including RESTful transports. The Basic Patient Privacy Consent (BPPC) profile is associated with this profile and these other existing profile. BPPC covers the legal and administrative needs for consent documentation and associating the patient consent with policy documentation. This profile includes the ability to associate the electronic authorizations with the patient agreements and organizational policies.

Greater integration of this authorization with third party authorization and consent documentation profiles, such as those found in the IHE BPPC profile, are a future effort. This profile starts with just the basic authorization activities.

It is important to understand that IUA is not a substitute for the administrative activities (such as withdrawing consent), policy setting, and other activities that BPPC documents.

The administrative actions needed to establish a third party as an authorization server for IUA is not covered by these actors or transactions. These activities are very much dependent upon the operational needs and privacy policies that apply to a particular deployment.

The IUA profile does convey the identifiers and signatures needed to establish traceability between the Authorized RESTful transaction and the policies and consents behind that authorization.

The HTTP RESTful transport is being used by many healthcare applications and smart devices. These share a common set of issues. A typical example is:

- The patient has a tablet and installs an application onto that tablet.
- An application will need to retrieve and update health related data that is stored on a resource server. It uses HTTP RESTful transactions for both retrieve and update because HTTP support is integrated into the platform services.
- The patient already has an established relationship with an authorization service.
- The patient wants to configure the application to have access to their data without needing the IT staff at the application vendor and resource vendor to set things up.

One common pattern is to interact directly with the application to communicate with the authorization service. The application interacts with both patient and authorization service to support the granting of an access token. The application then saves the access token, and uses it to retrieve and update the health related data. Another common pattern is for the user to interact independently with the authorization service and obtain a token. This token is saved on the device for later use.

The key issues here are:

- Reliable and accurate authorization decisions, as part of an overall privacy protecting and security environment.
- Application developers want one common method for obtaining and using these tokens, not thousands. They want a method that is built into the common platforms, not one that must be added later, because it is difficult for end user oriented applications to modify the platforms.
- Resource servers want one common method for receiving these tokens as part of HTTP RESTful transactions, and one common method for processing these tokens. (They will settle for a small number of methods if they must.)
- Users, patients and providers, want to be in control, do not want to depend on support staff to set up their devices and applications, and want to minimize the interference from authorization requirements.

Similar issues arise with:

- In house application distribution that needs to authorization for devices used within the facility.
    - The in house IT staff want a common method to authorize use of in house web applications and access to in house resources.
    - IT staff are more willing to run their own internal authentication and authorization servers, but want to use off the shelf software and want the option to outsource these services. They are more likely to separate authentication from authorization than end user systems. Authentication issues are closely related to HR activities like hiring and firing. Authorization issues are related to patient and work assignments. These are controlled by different parts of the organization and have different process dependencies.
    - Efficient user workflow requires minimizing the number of times a person is challenged for authentication by interactive applications.
- Providers and Specialists have authorization needs for dealing with other organizations.
    - Providers and specialists need to deal with hundreds of resource services. A provider panel of 10,000 patients will need hundreds of relationships with different specialists, labs, priors, and other providers.

- The providers and specialists struggle to maintain hundreds of different authentication and authorization relationships today. Their IT staff struggle to support at all these different relationships. Neither wants delays or problems that will impact patient care.
- Efficient user workflow requires minimizing the number of times a person is challenged for credentials for interactive applications.

- Granting subset access to specialized provider. E.g., read access to cardiac info to physical therapy organization, forbidding access to other data like reproductive health and addiction data.

There are also environmental assumptions made by this profile.

First, it is assumed that there will be multiple access control engines working together. The IUA activities are one part of a federated system. IUA will work in conjunction with other access control engines. For example, a glucose monitor may be authorized to have access to a patient's medical record. The expectation is that this will mean access to all of the glucose related information, which will include a variety of measurements and prescriptions. But, it is expected that if the device requests information about sexually transmitted disease diagnosis it will be rejected.

Second, this profile is operating in an environment where access consents are managed by BPPC or other mechanisms. IUA is not a substitute for documenting, establishing, and modifying these legal agreements. It is a method by which those agreements are enforced. For example, there will be a documented consent agreement between a patient and a provider that the provider will provide medical records to a healthcare proxy that is identified and authorized by the patient. BPPC is one way to document that agreement.

Open Issues and Question

Issue	Description
1	This profile does not specify the internal structure of "client_id". This is a major concern for operations and security management. But, OAuth does not provide a full specification for client_id. It just specifies its purpose. DICOM's equivalent information attributes are: Manufacturer, Model, Software Versions, and Serial Number. The OAuth client ID must identify the device, the application (including any necessary version information), the particular instance, and any other information needed to identify the client application uniquely. Registration of clients is a significant operational and security problem that is being postponed until there is more experience with problems in the field and reasonable solutions. There is known danger from spoofing of client_id. At this time, the method for assignment of client_id is not included in the profile. In the field there are a variety of methods being tried. Many depend upon physical distribution methods or out of band communications to manage the authentication problems.
2	This profile mandates support for JWT token format. It has an XUA SAML option defined by IHE for ease of integration with the IHE WS-Security environment. You may also use other token formats as part of a deployment.
3	Audit messages are only defined for clients that are also Secure Applications. There is no defined auditing for other clients.

4	This profile does not require client grouping with Secure Node or Secure Application because it is using the OAuth issuance rules for client_id, see the security consideration section. It assumes that the client_id management will deal with these security considerations in a manner similar to the certificate management assumptions made for TLS and other certificate users.

Closed Issues

Issue	Description	
8	This profile uses only the Authorization: header for conveying the authorization information. The parameter form is not prohibited but is not compliant with the profile.	
9	This profile does not explain the ways that some Resource Servers utilize HTTP redirects to automate some kinds of authorization activities. The actual HTTP transactions used for Obtain Authorization Token and Authorized RESTful Transaction are as defined within this profile. The other transactions are under the control of the Resource Server and its design. Is an IHE explanation of how this works needed, or is the extensive industry documentation and tutorials used in other fields sufficient? No.	

Standards Evaluation and Tradeoffs (Closed issue)

Standards considered:

- OAuth 1.x – These are a predecessor to the OAuth 2.0 framework, documented in various RFCs. They have widespread use experience. This showed that there inter-operability problems between different services, but that when a single authorization service was used it worked on many different user platforms for many different applications. The interoperability problem meant that an application would specify that it supported "vendor – Oauth 1.x". There could be several vendors on the list supported, but adding a new authorization vendor usually meant changing code.

  Removed from consideration. There were no reasons identified to disagree with the OAuth community choice to move to OAuth 2.0. There is not a significant installed base within healthcare that would benefit from preserving OAuth 1.x use. There is not a functional argument for OAuth 1.x.

  These profiles were also strongly tied to user interactions involving use of a browser.

  These restrictions lead to the development of OAuth 2.0. It was written as a framework, rather than a single standard, so that it can be adapted to different uses.

  OAuth 1.x is specifically for HTTP transactions.

- OAuth 2.0 – This is documented in RFC-6749. It defines a framework, where many details of things like codes must be separately specified. Multiple vendors have working implementations. Profiles are being developed. Some may be single vendor profiles, leaving the interoperability situation unchanged. Others may be multi-vendor profiles for

better interoperability.

OAuth 2.0 is specifically for HTTP transactions.

- EUA – This is an IHE profile based on Kerberos. It has been robust and is widely used in enterprise environments. It depends upon the ability to exchange shared secrets between servers using unspecified administrative paths. This is easy when the servers are all under control of a single enterprise. It has been used in cross enterprise environments and is robust, but it has not been popular. The administrative issues are probably the reason it has not been popular.

  EUA has bindings to multiple different protocols including HTTP.

- HTTP Password – This has been available for a long time, and is in use. It suffers from the usual issues with username-password systems. It has not been popular.

  Removed from consideration. HTTP Password has not been popular, and simple password based approaches like this have significant administrative problems. The failure to reach acceptance despite years of availability indicate that this will not be acceptable.

- HTTP SAML – This has been available for a long time and is in use. It has not been popular with consumer devices.

- LDAP Authentication – This is very similar to EUA, and differs primarily in administrative details. An EUA compliant device can usually be configured to use LDAP Authentication.

- Attribute Certificates – This is not an alternative transaction standard, but introduces considerations that may affect the standard selection and profiling plans. HTTP headers can be defined to include any kind of certificate. The HTTP SAML, the Browser SSO, and OAuth 2.0 all have some flexibility in this regard. Attribute certificates are a kind of certificate defined by RFC 3281. SAML Browser SSO can co-exist with OAuth 2.0. The OAuth emphasis has been much stronger in the consumer markets, and the Browser SSO has been stronger in the business to business markets. Possibilities include:

  1. Generate a XUA update to incorporate defined use of attribute certificates independently of this profile.
  2. Use attribute certs as part of the token selection for a medical OAuth 2.0.
  3. Do both 1) and 2) as coordinated efforts in two separate work items for ITI.

<< Initially a matrix of alternatives and considerations>>

Criterion/ Requirement	OAuth 1.x (*removed*)	OAuth 2.0	EUA (HTTP-Kerberos)	HTTP Password (*removed*)	HTTP SAML	LDAP Authentication	Attribute Certs.
Mindshare of Application Developers	**In use, but considered flawed with interoperability limitations.**	**Very high current mindshare. More consumer and small business friendly.**	**It's old. It is not widely accepted for cross-enterprise.**	**Considered too simplistic.**	**Not recognized as consumer oriented. Usually B2B.**	**Enterprises often uncomfortable exposing it for authN and authZ. (Overlaps with EUA)**	
Backwards compatibility with OAuth 1.x (Installed base consideration).	**Interoperability issues are experienced.**	**Not directly compatible.**	**Nil**	**Nil**	**Nil**	**Nil**	
Delegation to Application	**Non-standard extension**	**Within framework**	**Complex provisioning**	**Nil**	**Complex provisioning**	**Complex Provisioning**	
Delegation to 3rd Human	**Weak**	**Within framework**	**Nil**	**Nil**	**Complex provisioning**	**Nil**	
Delegation to 3rd Organization	**Weak**	**Within framework**	**Complex provisioning**	**Nil**	**Complex provisioning**	**Nil**	
Multiple identity providers	**Nil**	**Within framework, uncommon**	**Nil**	**n/a**	**Nil**	**Nil**	
Multiple authorization providers	**Weak Interoperability**	**Within framework, prime goal**	**Nil**	**n/a**	**Nil**	**Nil**	
Identity provider need not be the Authorization provider	**Nil**	**Within framework**	**Complex provisioning**	**n/a**	**Complex provisioning**	**Nil**	
HTTP(x) interface	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**No**	

Criterion/ Requirement	OAuth 1.x (*removed)*	OAuth 2.0	EUA (HTTP-Kerberos)	HTTP Password *(removed)*	HTTP SAML	LDAP Authentication	Attribute Certs.
specified							
Platform support in common platforms (iOS, Android)	**Yes**	**Yes**	**No**	**No**	**No**	**No**	
Sharable certificate store	**n/a**	**n/a**	**n/a**	**n/a**	**n/a**	**n/a**	
Sessions	**Unspecified**	**Within framework**	**Yes**	**Unspecified**	**Unspecified**	**n/a**	
Revocable Delegation	**Nil**	**Within framework, uncommon**	**Yes**	**n/a**	**n/a**	**n/a**	
Short-term Delegation (hours)	**Nil**	**Within framework, uncommon**	**Yes**	**n/a**	**n/a**	**n/a**	
Long-term Delegation (weeks-months)	**Nil**	**Within framework, uncommon**	**Yes**	**n/a**	**n/a**	**n/a**	
Bi-directional TLS Authentication of Node	**Nil**	**Within framework, serious implementation issue.**	**Yes**	**Yes**	**Yes**	**n/a**	
Uni-directional TLS Authentication of Server Node	**Yes**	**Within framework, implementation issue**	**Yes**	**Yes**	**Yes**	**Yes**	
Multiple kinds of authorization over one connection? Require resource per	**No**	**Within framework, uncommon**	**Yes**	**No**	**No**	**n/a**	

Criterion/ Requirement	OAuth 1.x (*removed*)	OAuth 2.0	EUA (HTTP-Kerberos)	HTTP Password (*removed*)	HTTP SAML	LDAP Authentication	Attribute Certs.
kind of authorization.							
Authorization for classes of documents.	**?**	**Within framework, must be profiled**	**No**	**No**	**Yes**	**No**	
Support of policy expressiveness in the token **Indication of LOA used as part of transaction.** **Expiration mechanism**	**?**	**Within framework, must be profiled**	**No**	**No**	**Yes**	**No**	

Description of Criterion for evaluation

- Mindshare of Application Developers

  This criterion reflects the readiness of the mobile device platforms for the chosen technology. Application developers prefer to minimize the number of dependencies that they have beyond the basic platform. It also reflects the adoption by major non-healthcare Internet players like Google, Facebook, etc. The ideal for mindshare is something that is already part of the basic platform for IOS, Android, and is used by Google, Facebook, and others.

- Backwards compatibility with OAuth 1.x (Installed base consideration).

- Delegation to Application

  An un-attended application must be able to be given authorization to gain access to healthcare data. Automatic applications like glucose monitors will not be usable if a patient interaction is needed whenever data is to be transferred. The patient needs to be able to set up the device and forget about it.

- Delegation to 3rd Human

  A patient needs to be able to delegate authorization to another person. This might be for purposes of a healthcare proxy while the patient is unable to make decisions. It might be so that an advisor can obtain information.

- Delegation to 3rd Organization

A patient needs to be able to delegate authorization to an organization. This might be so that a doctor has access to records, or a visiting nurse organization has access to records.

- Support Multiple identity providers

A provider will sometimes need to be able to accept identities from multiple identity providers. A patient will sometimes need to have identities that are managed by multiple identity providers.

- Support Multiple authorization providers

A provider will sometimes need to be able to accept authorizations from multiple authorization providers. A patient will sometimes need to deal with multiple authorization providers.

- Identity provider need not be the Authorization provider

The identity provider will sometimes be the same as the authorization provider. In these cases, authorization and identification are often combined into a single simplified set of transactions. In other cases, the identity provider and authorization provider will be separate servers. This separate case is presently more common in the enterprise environments, where the identity provision is related to HR functions and the authorization is related to the persons current work assignments. The selected approach must work for both environments.

- HTTP(x) interface specified

This service must use HTTP framing (e.g., HTTP headers) only. It may also require use of protective security layers like TLS, but must not require modification of those layers. It may use other protocol layers for ancillary services, but mobile devices cannot be expected to provide those as platform services. Such use is a definite negative.

- Platform support in common platforms (iOS, Android)

The selected standards must be able to be used on the common platforms of IOS, Android, Windows, MacOS, and Linux.

- Sharable certificate store

The selected standards should not interfere with use of shared certificate stores. Some platforms enable shared storage of certificates and other security related tokens. The standards should not prevent use of shared stores.

- Sessions

Many healthcare applications need to perform multiple HTTP transactions as part of basic function. The selected standard should not require a full re-authorization for every HTTP transaction. Some sort of token or session mechanism will be needed for performance reasons.

- Revocable Delegation

  The authorization delegations to devices, people, and organizations must be revocable. A revision method is not as critical, since revision can be done as a revocation and new authorization. The revocation must not require access to the device, since devices can be lost and stolen.

- Short-term Delegation (hours)

  Some delegations are known to be needed for only a short time. There should be a way to authorized for only a limited number of hours.

- Long-term Delegation (weeks-months)

  Some delegations are known to be needed for a long time. There should be a way to authorize for a long period. E.g., a blood pressure monitor should be able to be authorized to report blood pressure for months or years.

- Bi-directional TLS Authentication of Node

  Some uses, e.g., physician in-house tablet, need full bi-directional authentication of the device. This should be supported.

- Uni-directional TLS Authentication of Server Node

  It will not always be practical to fully authenticate each patient device. The authentication service must not force such authentication. It is allowed to permit authentication of the server side only.

- Multiple kinds of authorization over one connection? Require resource per kind of authorization?

  Access to some kinds of healthcare data requires multiple authorizations. Mechanisms that support this are acceptable but not required. These situations are not common in the mobile environment.

- Authorization for classes of documents.

  Authorizations need to be available for a class of resources or documents. These classes will be defined by policy and agreement. For example, a doctor reviewing diabetic care should be able to retrieve all relevant documents based upon a single authorization, rather than require re-authorization for each individual document and resource.

- Support of policy expressiveness in the token

  The authorization token may have the ability to include significant self-description. For example, it may be able to include a string saying “Authorization based on BPPC document with UID x.y.z”.

- Indication of level of assurance (LOA) used as part of transaction.

The authorization token may indicate a LOA that was performed in generating the token. The server may specify a desired LOA for the tokens that it will accept.

General Introduction

Update the following Appendices to the General Introduction as indicated below. Note that these are not appendices to Volume but rather to the General Introduction.

Appendix A - Actor Summary Definitions

Add the following actors to the IHE Technical Frameworks General Introduction list of Actors:

Actor	Definition
Client Authorization Agent	A RESTful client that provides authorization information for RESTful transactions.
Authorization Server	A server that provides authorization tokens
Resource Server	A server that provides RESTful services that need authorization

Appendix B - Transaction Summary Definitions

Add the following transactions to the IHE Technical Frameworks General Introduction list of Transactions:

Transaction	Definition
Request Authorized Service	A RESTful transaction that incorporates authorization information and controls
Obtain RESTful Authorization	A transaction that is used to provide an authorization token for use in Authorized RESTful transactions.

Volume 1 – Profiles

X IUA Profile

The IUA profile adds authorization information to RESTful transactions. The IUA actors and behavior will be added to other profiles and transactions that need authorization.

X.1 IUA Actors, Transactions, and Content Modules

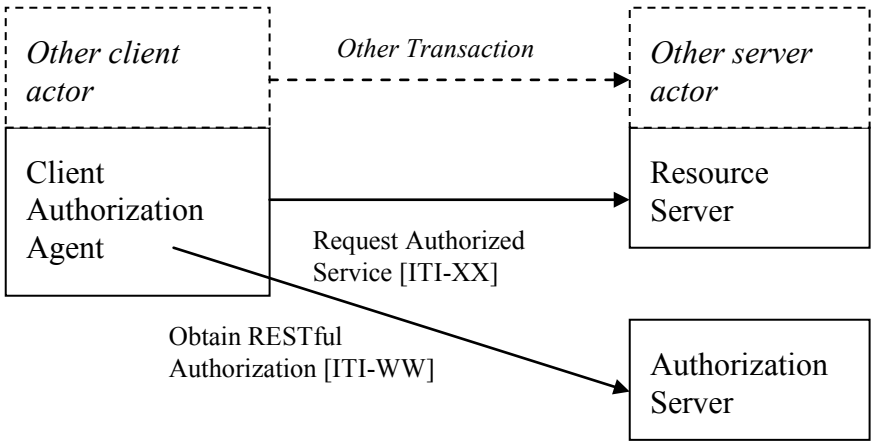

Figure X.1-1: IUA Actor Diagram

Table X.1-1 lists the transactions for each actor directly involved in the IUA Profile. To claim compliance with this Profile, an actor shall support all required transactions (labeled "R") and may support the optional transactions (labeled "O").

Table X.1-1: IUA Profile - Actors and Transactions

Actors	Transactions	Optionality	Reference
Client Authorization Agent	Request Authorized Service	R	ITI TF-2: 3.Y1
	Obtain RESTful Authorization	O	ITI TF-2: 3.Y2

Actors	Transactions	Optionality	Reference
Authorization Service	Obtain RESTful Authorization	R	ITI TF-2: 3.Y1
Resource Service	Request Authorized Service	R	ITI TF-2: 3.Y3

X.1.1 Actor Descriptions and Actor Profile Requirements

The IUA actors are expected to be grouped with other actors that perform RESTful transactions. Grouping an Authorization Client with another actor means that this other actor will provide an authorization token as part of the HTTP transaction to a RESTful server. It may perform the Obtain RESTful Authorization transaction to obtain the authorization token. The corresponding RESTful server should be grouped with the Resource Service actor to indicate that the server will perform access control using the OAuth token and transaction rules.

X.1.1.1 Client Authorization Agent

The client authorization agent actor performs the network transactions and user interactions needed to obtain an authorization token and to attach that token to HTTP RESTful transactions so that those transactions are known to be authorized.

This activity has two transactions:

- The Request Authorized Service transaction – In this case the authorization token has already been obtained and is communicated as part of the HTTP RESTful transaction for some other profile or service. This token indicates that the RESTful transaction has been authorized for a particular kind of service and particular device by an authenticated person.
- The Obtain RESTful authorization – In this use, the authorization client interacts with an Authorization service, Integrated Authorization Service, and Authentication Service as needed to obtain a token that indicates RESTful transactions for a particular kind of service and device are authorized by a particular person. This will often include various interactions with the user for authentication purposes. Those interactions are outside the scope of this profile, and may involve biometric or other identification activities. The resulting token is saved for later use by the authorization client and must be protected.

X.1.1.2 Authorization Server

The Authorization Server uses an authenticated user identity and the requested RESTful service URL to determine whether RESTful transactions are allowed. If they are allowed, the authorization service provides a token indicating the user identity and RESTful service access that are permitted.

X.1.1.3 Resource Server

The Resource Service accepts a RESTful transaction with an authorization token attached. It evaluates the authorization token to verify that this is an authorized transaction. It then allows the transaction to proceed, subject to other constraints that may also be in place.

Note: For implementation and deployment reasons the Resource Server and Authorization Server can be combined into an integrated product together with user authentication, access control, and other services. This does not change the actor requirements or transactions used.

X.2 IUA Actor Options

Table X.2-1: IUA - Actors and Options

Actor	Option Name	Reference
Authorization Server	XUA SAML Token	X.2.2
Resource Server	XUA SAML Token	X.2.2

X.2.1 XUA SAML Token Option

A Client Authorization Agent or Authorization Service that claims the XUA SAML Token option shall be able to use or generate the SAML tokens defined in the XUA option as the access token for IUA.

X.3 IUA Required Actor Groupings

An Authorization Server shall be grouped with the Time Client Actor in Consistent Time. A Resource Service shall be grouped with the Time Client Actor in Consistent Time.

This profile does not require client grouping with Secure Node or Secure Application because it is using the OAuth issuance rules for client_id, see the security consideration section.

X.4 IUA Overview

X.4.1 Concepts

The term “authorization” and “access control” are used colloquially for a variety of related activities. This profile will use more specific terms for each of these activities. These are:

- Provisioning – Setting up the initial rules and updating them when the situation changes. The administrator may say “Authorize Dr. X to have access”. The steps taken to make this happen are called provisioning.
- Delegation – Adding, transferring and revoking authorization from one person to another. This is closely related to provisioning. It differs in that it can only transfer authority that

has already been provisioned, and it may track changes to provisioned access for the original person.

- Authentication – Determining that the actual user (at the moment of authentication) is a known identity.
- Authorization – Determining that the authenticated user is authorized to have access to a resource (at the moment of authorization). The profile describes how to convey an access authorization decision. It is not defining how the decision is made.
- Access Control – A system of provisioning, delegation, authentication, and authorization. It is normal to have multiple nested levels of access control. This profile is concerned with access control at the HTTP level. There are likely also building access controls, resource server access controls, and other access controls involved.

Within this profile, authorization is limited to the definition of authorization above.

X.4.2 Use Cases

There is one primary use case for authorization for access to a resource, which takes four different common variations. There is also one delegation use case that is perhaps in scope for this profile. There are other use cases for delegation, provisioning, etc. that are out of scope for this profile.

The primary use case is that a user desires access to a resource using HTTP mechanisms. The user might be a patient, a provider, a machine, a software application, or some other kind of participant.

The authorization service is provided by a different organization or part of the organization than the resource service. This need is driven by the requirements of patients, providers, and other users to simplify and maintain autonomy and control over authorization services. A user may interact with dozens of providers. It is difficult for the user to coordinate different authorization mechanisms with each of these dozens of providers.

In other kinds of Internet usage there are vendors of authorization services that are used to solve this problem. These include Facebook, Google, and a variety of other service providers from different commercial and governmental sectors. These overlap with providers of authentication services. For both authentication and authorization these services allow a patient to establish an authentication and authorization system with minimal provisioning by the healthcare provider. The user can specify “use vendor X” to their provider.

The pre-requisites for these use cases are:

- The User has established a relationship with the Authentication and Authorization services.
- The resource service has agreed to use the same Authentication and Authorization services. This is often much easier than establishing and maintaining their own patient facing authentication and authorization services. The agreement to use an external

service is a significant policy choice, because it is accepting some shared responsibility for choosing suitable authentication and authorization services. The user shares part of this decision responsibility, but local laws and regulations will affect a resource servicer's decision to accept and use a third party authorization and authentication service.

- The authentication and authorization services have agreed to be used by the User and resource service provider.

X.4.2.1 Simple Authorization

A provider with a mobile device wishes to retrieve a medical document to which they have authorized access.

The User communicates first with the authentication and authorization services to obtain a token that will be presented to the resource service. This token will be used as part of an access control decision by the resource service.

The User could be any kind of participant, and the resource use could be retrieval, query, or storage of a resource by means of HTTP transactions.

X.4.2.2 Machine Delegation

There are three significant and common reasons to perform delegations. These cases are primarily patient delegation choices. Providers rarely have the authority to delegate. IT staff may use delegation as part of the support for autonomous devices.

Users may delegate authority to:

- Device or applications that are performing a service for the patient, for example automatic glucose monitors that can provide monitoring records and receive control information from a healthcare provider service that is providing diabetic care.
- Applications that are distributed across multiple devices, multiple instantiations. E.g., Kindle devices synchronize last read location, documents available, etc. across multiple Kindle devices for a single user account.
- Advocates and proxies who are authorized by the patient to make decisions for the patient.
- Organizations that are acting for the patient, such as a visiting nurse organization that is providing support to the patient.

Revocation needs to be clearly specified by policy. Revocation may be removal of rights because of swapping devices. Expiration, re-authorization, etc. also need to be covered. Revocation is not just a response to breaches and failures. Revocation is a normal response to changes in people, equipment, and relationships.

Only the first use case is considered in scope for the IUA profile. The multiple device synchronization, human advocates, and organizational relationships involve a great many rules, regulation, policies, and procedural variations. These are all outside the reasonable scope of an

IHE profile. These local policies and procedures might take advantage of IUA for part of the policy or procedure.

X.4.2.2.1 Obtaining a token for an autonomous device

Autonomous devices like patient monitors can use the Request Authorized Service transactions without using the Obtain Authorization transaction. These machines often require special software and connections as part of their configuration process. Often this process is done using a PC or other system communicating with the device by USB or Bluetooth. A device specific application handles the various device specific configuration setup details for a particular patient. An appropriate authorization token can be provided as part of this configuration process. It can then be used for Request Authorized Service transactions.

The device must meet the requirements of being an OAuth confidential client for this to be secure.

Other autonomous systems within the IT environment can also be configured similarly, provided they meet the requirements for being an OAuth confidential client. In this case the setup is usually performed by the IT staff as part of system configuration.

In all of these cases, the authorization token identifies the device that is being authorized to perform the RESTful transaction and the patient involved, so that the appropriate access control decisions can be made.

X.5 IUA Security Considerations

The OAuth RFC has references to some relevant security analyses. There are also a wide variety of analyses in the public literature. This profile does not introduce new considerations to those analyses. We have not identified any new healthcare related issues.

It is important to understand that IUA does not address the issues around issuing and revoking client_ID's. OAuth 2.0 depends upon the client_ID to establish the degree of trust in a client. OAuth 2.0 does not define further how client_ID's are managed.

There are significant administrative issues dealing with establishing the appropriate level of trust with client applications, vendors, etc. These also include establishing methods for dealing with the discovery of flaws, breaches, etc. These affect both the Resource Server and Authorization Server administrative support.

The Authorization Server will have an administratively managed list of approved client_ids for acceptable clients. This list will be updated as new clients are approved or existing clients are removed. An authorization token will not be issued for unapproved clients. This assumes that the client_id management will deal with these security considerations in a manner similar to the certificate management assumptions made for secure communication transactions.

The Resource Server may also have such a list if there is a more precisely managed list of client_id and resource content access requirements. This can deal with resources that have more specific client requirements than the general access authorization requirements.

X.6 IUA Cross Profile Considerations

None.

Volume 2 – Transactions

Add section 3.W

3.W Obtain RESTful Authorization

3.W.1 Scope

This transaction is used to obtain the access token for use in a RESTful Resource request.

3.W.2 Actor Roles

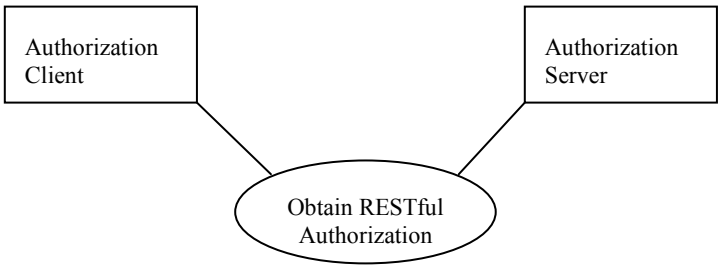

Figure 3.W.2-1: Use Case Diagram

Table 3.W.2-1: Actor Roles

Actor:	Authorization Client
Role:	Authorization portion of a RESTful transaction client.
Actor:	Authorization Server
Role:	Server that grants access tokens

3.W.3 Referenced Standards

- RFC-6749 OAuth 2.0 Authorization Framework
- RFC-6750 OAuth 2.0 Authorization Framework: Bearer Token Usage
- RFC-draft JSON Web Token (JWT) *draft-ietf-oauth-json-web-token*
- RFC-draft JSON Web Token (JWT) Bearer Token Profiles for OAuth 2.0 *draft-ietf-oauth-jwt-bearer*

- RFC-draft SAML 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants *draft-ietf-oauth-saml2-bearer*

3.W.4 Interaction Diagram

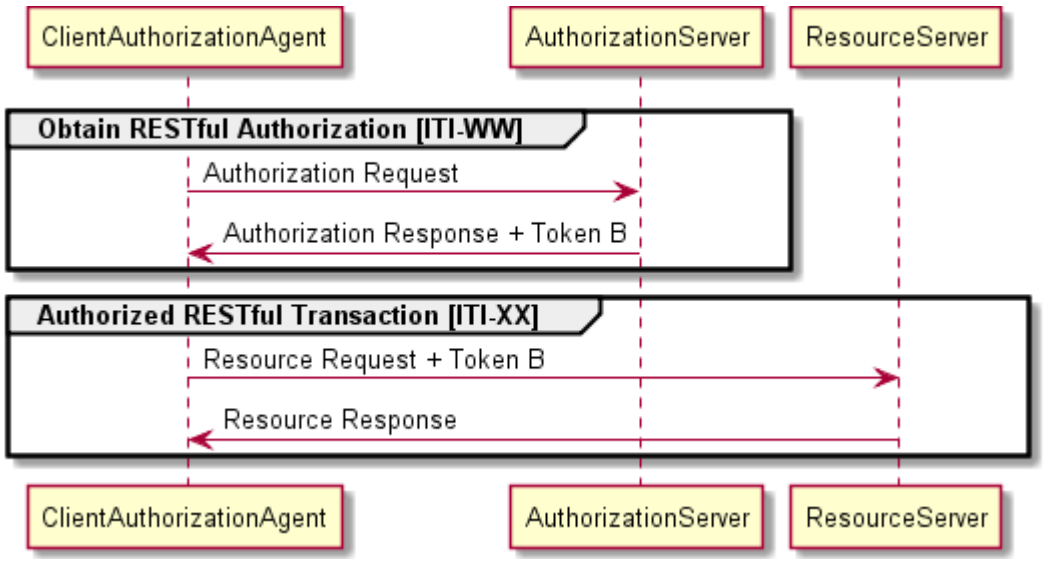

Figure 3.W.4-1: Basic Process Flow for Obtain Restful Authorization and Request Authorized Service Transaction

Pre-conditions:

Main Flow:

1. The user provides user authentication and the intended resource request information to the authorization server.
2. The authorization server generates a token B that indicates that this user is authorized to have access to this resource.

Post-conditions:

The device moves on to the Request Authorized Service Transaction

Note: There are other means by which a device can get an authorization token B. Some devices may be configured by device specific methods with an appropriate token. A token might be generated from an XUA SAML assertion.

3.W.4.1 Authorization Request

The Authorization request is an HTTP GET transaction used to obtain an authorization token that will be used for subsequent RESTful transactions.

3.W.4.1.1 Trigger Events

This transaction takes place whenever an Authorization Client needs an access token authorizing a RESTful transaction. This may be due to expiration of an existing token, a resource request has indicated that a new token is required, configuration or installation of a device, or as a routine request for new transactions.

3.W.4.1.2 Message Semantics

3.W.4.1.2.1 JWT Token

The Authorization Client and Authorization server shall support the JWS (signed) alternative of the JWT token. Any actor that supports the IUA may support the JWE (unsigned but encrypted) alternative of the JWT token.

The JWT token attribute requirements are shown in table-3.W.4.1.2.1. The required attributes are indicated by "R". Optional attributes are indicated by "O". If present, the optional attributes shall be used in accordance with OAuth and JWT specifications.

Table 3.W.4.1.2.1-1: JWT Token requirements

Parameter	Req	Definition	RFC Reference
iss	R	Issuer of token	Draft json-web-token Section 4
sub	R	Subject of token (e.g. user)	Draft json-web-token Section 4
aud	R	Audience of token	Draft json-web-token Section 4
exp	R	Expiration time	Draft json-web-token Section 4
nbf	O	Not before time	Draft json-web-token Section 4
iat	O	Issued at time	Draft json-web-token Section 4
typ	O	Type	Draft json-web-token Section 4
jti	R	JWT ID	Draft json-web-token Section 4

The Authorized Client, Authorization Server, and Resource Server shall support the following extensions to the JWT parameters. All of these parameters are optional in the JWT token. The parameter content shall be the same as the content defined in ITI-40. The definition is summarized in this table for convenience.

Table 3.W.4.1.2.1-1: Extensions to JWT Parameters

XUA Attribute	XUA Definition	JWT Parameter
SubjectID	Plain text user's name	SubjectID
SubjectOrganization	Plain text description of the Organization	SubjectOrganization
SubjectOrganizationID		SubjectOrganizationID

XUA Attribute	XUA Definition	JWT Parameter
HomeCommunityID	Home Community ID where request originated	HomeCommunityID
NationalProviderIdentifier		NationalProviderIdentifier
Subject:Role		SubjectRole
docid	Patient Privacy Policy Acknowledgement Document ID	docid
acp	Patient Privacy Policy Identifier	acp
PurposeOfUse	Purpose of Use for the request	PurposeOfUse
Resource-ID	Patient ID related to the Patient Privacy Policy Identifier	resourceID

3.W.4.1.2.2 XUA SAML Token Option

Any actor claiming conformance with the XUA SAML Token option shall comply with the SAML 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants (RFC- *draft-ietf-oauth-saml2-bearer*) rules for issuing and using SAML assertions and tokens. The SAML assertion contents shall comply with XUA SAML assertion rules (see ITI TF-2b:3.40). All of the XUA options shall be supported.

3.W.4.1.3 Expected Actions

The specific HTTP transactions are defined in the referenced standards. This profile does not modify them other than through the definition of additional token attribute rules and auditing requirements. The end result of those transactions will be either an error response, as defined in the RFCs, or an access token that can be used in the Request Authorized Service Transaction.

3.W.5 Security Considerations

3.W.5.1 Security Audit Considerations

3.W.5.1.1 Resource Server Specific Security Considerations

The Resource Server shall generate an audit message when an authorized transaction is performed or attempted.

	Field Name	Opt	Value Constraints
Event	EventID	M	EV(tdb, tbd, “Authorization”)

	EventActionCode	M	"E" (Execute)
	EventDateTime	*M*	*not specialized*
	EventOutcomeIndicator	*M*	*not specialized*
	EventTypeCode	M	EV(tbd, tbd, "OAuth Authentication")
Source (1)			
Human Requestor (0)			
Destination (0)			
Audit Source (Client Authentication Agent) (1)			
Participant Object (1)			

Where:

Source AuditMessage/ ActiveParticipant	UserID	M	The process ID as used within the local operating system in the local system logs.
	AlternativeUserID	U	*not specialized*
	UserName	U	*not specialized*
	UserIsRequestor	M	*not specialized*
	RoleIDCode	M	EV(110150, DCM, "Application")
	NetworkAccessPointTypeCode	M	"1" for machine (DNS) name, "2" for IP address
	NetworkAccessPointID	M	The machine name or IP address, as specified in RFC 3881.

Audit Source AuditMessage/ AuditSourceIdentification	*AuditSourceID*	*U*	*Not specialized.*
	AuditEnterpriseSiteID	*U*	*not specialized*
	AuditSourceTypeCode	*U*	*not specialized*

640

Token (AuditMessage/ ParticipantObjectIdentification)	ParticipantObjectTypeCode	M	"2" (System)
	ParticipantObjectTypeCodeRole	M	"13" (Security Resource)
	ParticipantObjectDataLifeCycle	U	*not specialized*
	ParticipantObjectIDTypeCode	U	*not specialized*
	ParticipantObjectSensitivity	U	*not specialized*
	ParticipantObjectID	U	*not specialized*
	ParticipantObjectName	U	*not specialized*
	ParticipantObjectQuery	M	*URL requested*
	ParticipantObjectDetail	M	*IP address of requesting system*

Add section 3.X

3.X Request Authorized Service Transaction

3.X.1 Scope

This transaction is used to provide authorization information as part of a RESTful transaction. This transaction specified some headers and behavior that must be part of a RESTful transaction. The rest of RESTful transaction specification for the URL, parameters, other headers, and other transaction contents is in another profile or specification.

3.X.2 Actor Roles

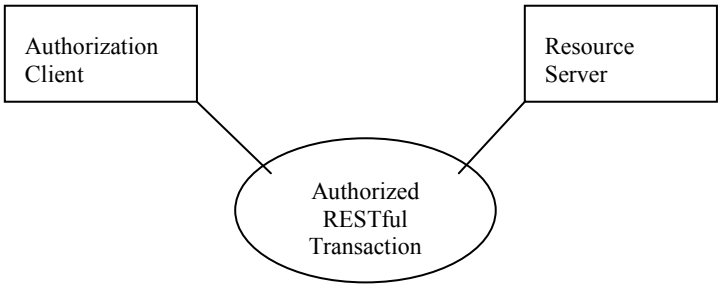

Figure 3.X.2-1: Use Case Diagram

Table 3.X.2-1: Actor Roles

Actor:	Authorization Client
Role:	Authorization portion of a RESTful transaction client.
Actor:	Resource Server
Role:	Authorization portion of a RESTful transaction server.

3.X.3 Referenced Standards

- RFC-6749 OAuth 2.0 Authorization Framework
- RFC-6750 OAuth 2.0 Authorization Framework: Bearer Token Usage
- RFC-draft JSON Web Token (JWT) *draft-ietf-oauth-json-web-token-07 (or most recent)*
- RFC-draft JSON Web Token (JWT) Bearer Token Profiles for OAuth 2.0 *draft-ietf-oauth-jwt-bearer*
- RFC-draft SAML 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants *draft-ietf-oauth-saml2-bearer*

3.X.3.1 Related IHE Profiles

XUA Cross-Enterprise User Assertion – Attribute Extension

3.X.4 Interaction Diagram

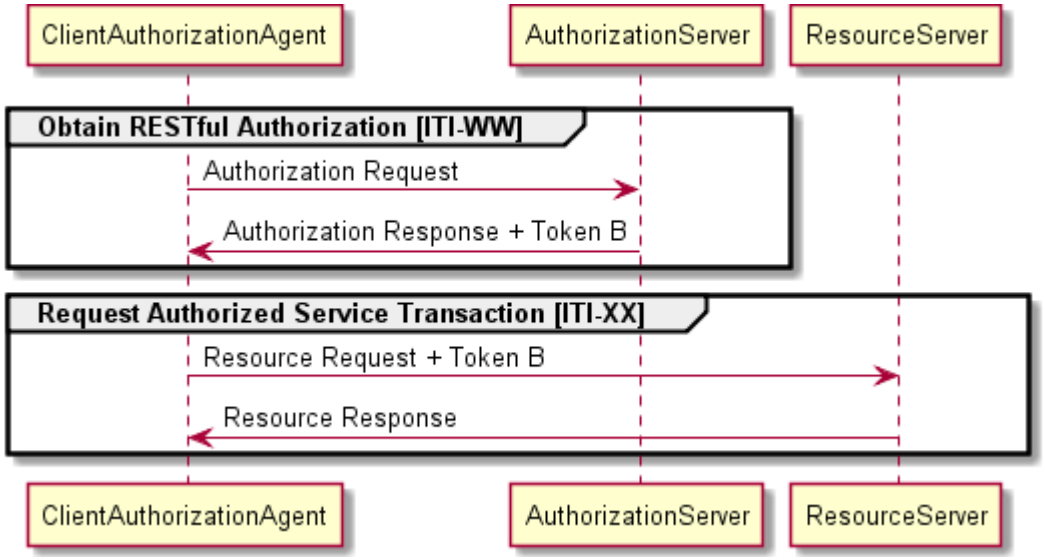

Figure 3.X.4-1: Process flow for Request Authorized Service Transaction

Main Flow:

1. The device sends a resource request to the resource server, together with the authorization token B. Token B may be an XUA SAML token, a JWT Bearer token, or another token type that is mutually agreed between Client, Resource Service and the token source.
2. The resource service provider makes an access control decision based upon the user identity, authorization token B, and resource requested. It may provide the resource, a subset of the resource, or reject the request.

Note: The token source in the diagram is not necessarily an IHE Actor. It is any system that provides an authorization token. It can be the Authorization Server, or it can be some other system.

This transaction works in conjunction with some other RESTful transaction. It extends the other transaction by adding information to the HTTP request for that other RESTful transaction. This is usually the addition of an authorization token to the HTTP request URL.

3.X.4.1.1 Trigger Events

The client system needs to make a RESTful transaction to a Resource Server that performs access authorization. The Authorization client has already obtained the necessary access token, either by means of another IHE transaction or by some other means.

3.X.4.1.2 Message Semantics

The Authorization Client should:

1. Confirm that the access token is still valid. Attempts to communicate using an expired token will result in an error.
2. Include an `Authorization:` header in the HTTP transaction that has the access token value. See RFC 6750 section 2.1. Further fields in the Authorization: header depend upon the token option chosen. The access token may be:
   - A JWT token, encoded as defined below, with the attribute requirements defined below.
   - A SAML token encoded as described in the option below, for compatibility with the XUA option.
   - A token of another type.

```
GET /example/url/to/resource/location HTTP/1.1
Authorization: IHE-JWT fFBGasru1FQd[…omitted for brevity…]44sdfAfgTa3Zg
Host: examplehost.com
```

The remainder of the transaction requirements are established by the RESTful transaction being protected.

Note: The draft RFCs have not specified the authorization code yet. Until there are official codes assigned, IHE will use IHE-JWT.

3.X.4.1.2.1 SAML token option

An Authorization Client that supports the SAML Token option shall be able to accept and use a SAML assertion that complies with the XUA specification as the access token for this request. A Resource Server that supports the SAML Token option shall be able to accept and use a SAML assertion that complies with the XUA specification as the access token for a request.

The SAML assertion shall be encoded as specified by SAML 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants (RFC- *draft-ietf-oauth-saml2-bearer*). This shall be included in the HTTP headers as an Authorization of type IHE-SAML.

```
GET /example/url/to/resource/location HTTP/1.1
Authorization: IHE-SAML fFBGRNJru1FQd[…omitted for brevity…]44AzqT3Zg
Host: examplehost.com
```

Notes: 1. WS-Trust defines methods for converting between SAML and JWT tokens. This profile does not specialize or change those methods.

2. The draft RFCs have not specified the authorization code yet. Until there are official codes assigned, IHE will use IHE-SAML.

The Resource Server should return an HTTP 401 (UnAuthorized) error if the token is not accepted. The other actor that is grouped with the Resource Server will not process the request in any way.

If the token is accepted, the other RESTful service actor will then process the request. This other service may generate success or error responses for reasons established by that other service. The other service may apply further context dependent access controls, restrictions on data returned, etc. A valid token only authorizes access to the other service. Information within the token may be used in these subsequent decisions.

Note: It is possible that the other service might also return a 401 (UnAuthorized) error. It is possible, although not recommended, that the Resource Server will choose a different error code for invalid, expired, or otherwise unacceptable access tokens.

3.X.5 Security Considerations

3.X.5.1 Security Audit Considerations

3.X.5.1.1 Resource Server Specific Security Considerations

The Resource Server shall generate an audit message for every authorization transaction.

Where:

	Field Name	Opt	Value Constraints
Event AuditMessage/ EventIdentification	EventID	M	EV(tdb, tbd, "Authorization")
	EventActionCode	M	"E" (Execute)
	EventDateTime	*M*	*not specialized*
	EventOutcomeIndicator	*M*	*not specialized*
	EventTypeCode	M	EV(tbd, tbd, "OAuth Authentication")
Source (1)			
Human Requestor (0)			
Destination (0)			
Audit Source (Client Authentication Agent) (1)			
Participant Object (1)			

Where:

Source AuditMessage/ ActiveParticipant	UserID	M	The process ID as used within the local operating system in the local system logs.
	AlternativeUserID	U	*not specialized*
	UserName	U	*not specialized*
	UserIsRequestor	M	*not specialized*
	RoleIDCode	M	EV(110150, DCM, "Application")
	NetworkAccessPointTypeCode	M	"1" for machine (DNS) name, "2" for IP address
	NetworkAccessPointID	M	The machine name or IP address, as specified in RFC 3881.

Audit Source AuditMessage/ AuditSourceIdentification	*AuditSourceID*	*U*	*Not specialized.*
	AuditEnterpriseSiteID	*U*	*not specialized*
	AuditSourceTypeCode	*U*	*not specialized*

750

Token (AuditMessage/ ParticipantObjectIdentification)	ParticipantObjectTypeCode	M	"2" (System)
	ParticipantObjectTypeCodeRole	M	"13" (Security Resource)
	ParticipantObjectDataLifeCycle	U	*not specialized*
	ParticipantObjectIDTypeCode	U	*not specialized*
	ParticipantObjectSensitivity	U	*not specialized*
	ParticipantObjectID	U	*not specialized*
	ParticipantObjectName	U	*not specialized*
	ParticipantObjectQuery	M	*URL requested*
	ParticipantObjectDetail	M	*IP address of requesting system*

