OpenID Connect as a Security Service in Cloud-based Diagnostic Imaging Systems

Weina Ma[1], Kamran Sartipi[1], Hassan Sharghi[1], David Koff[2], Peter Bak[3]
[1]Department of Electrical, Computer and Software Engineering,
University of Ontario Institute of Technology, Oshawa, ON L1H 7K4, Canada
[2]Department of Radiology, McMaster University, Hamilton, ON, L8S 4L8, Canada
[3]Medical Imaging Informatics Research Centre at McMaster (MIIRC@M), Hamilton, ON, Canada

ABSTRACT

The evolution of cloud computing is driving the next generation of diagnostic imaging (DI) systems. Cloud-based DI systems are able to deliver better services to patients without constraining to their own physical facilities. However, privacy and security concerns have been consistently regarded as the major obstacle for adoption of cloud computing by healthcare domains. Furthermore, traditional computing models and interfaces employed by DI systems are not ready for accessing diagnostic images through mobile devices. RESTful is an ideal technology for provisioning both mobile services and cloud computing. OpenID Connect, combining OpenID and OAuth together, is an emerging REST-based federated identity solution. It is one of the most perspective open standards to potentially become the de-facto standard for securing cloud computing and mobile applications, which has ever been regarded as "Kerberos of Cloud". We introduce OpenID Connect as an identity and authentication service in cloud-based DI systems and propose enhancements that allow for incorporating this technology within distributed enterprise environment. The objective of this study is to offer solutions for secure radiology image sharing among DI-r (Diagnostic Imaging Repository) and heterogeneous PACS (Picture Archiving and Communication Systems) as well as mobile clients in the cloud ecosystem. Through using OpenID Connect as an open-source identity and authentication service, deploying DI-r and PACS to private or community clouds should obtain equivalent security level to traditional computing model.
Keywords: Diagnostic imaging, PACS, identity service, authentication service, OpenID Connect, cloud

1. INTRODUCTION

Cloud computing is rapidly emerging as a preferred solution for information sharing over the Internet using external infrastructure, and allows access to applications and data on demand, any time, from anywhere. Although being a late arrival on the scene, the healthcare communities have started to consider the use of cloud technologies and deployment models for delivering timely and effectively IT-enabled healthcare services. By taking advantage of the cloud capabilities, the healthcare systems would benefit a lot both from allowing organizations to deploy applications without constraining to their own physical facilities, and from allowing organizations to deliver better healthcare services to patients using the spectrum of health service providers.

In medical imaging, PACS (Picture Archiving and Communication System) is complex integrated system equipped with the necessary hardware and software: digital imaging acquisition devices namely modalities (e.g., CT scanner, MRI system); digital image storage and archive where the acquired images are stored; and workstations where radiologists view the images.[1] With the increasing demand of collaborative work and sharing of medical information, PACS systems in different hospitals or image centers are interconnected across distributed environment. Diagnostic imaging repository (DI-r) provides a solution for sharing (publishing, discovery, retrieving and reliably storing) of DI documents across affiliated healthcare organizations. According to the status of DI-r projects across Canada[2], provincial DI-r's have been developed to deliver fast and easy access to diagnostic images to all authorized healthcare providers.

With the exploding rate of DI data and the fast growth in DI market, migrating PACS systems and DI-r services to cloud platform is cost-effective and improves the quality of DI services. However, it is a big challenge of managing the identity of various participants (e.g., user, device, application), and make sure all service provides can provide authentication mechanism with secure equivalents in cloud ecosystem. With the shift of federated identity solutions, from being organization-centric to user-centric, account information is persisted and managed by third party services and the users are authenticated by co-operating sites (e.g., PACS and DI-r services) using these services. OpenID is an open standard user-centric SSO (Single Sign-On) solution, which is rapidly gaining adoption on the web services, with over

one billion OpenID enabled use accounts and over 50,000 websites accepting OpenID for logins.[3] "OpenID Connect" is an identity layer on top of the OAuth 2.0 protocol produced by OpenID Foundation. OpenID Connect is the next generation of OpenID, but with more friendly-API, and is usable by native and mobile applications.[3] According to the OpenID Foundation, the US government has collaborated with OpenID Foundation to build open trust frameworks for citizens to easily and safely engage with government websites.[3] Also the department of health and human services of US Government has joined the OpenID Foundation to create a profile of OpenID Connect and associated pilot projects.[3]

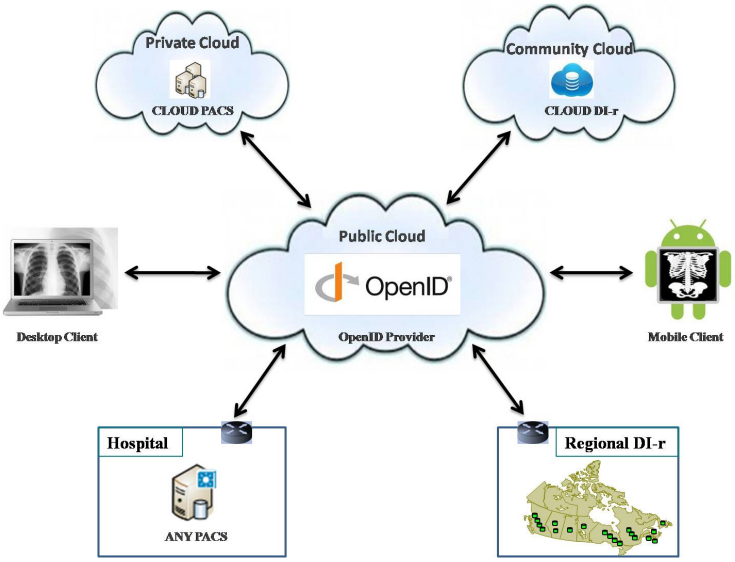

Figure 1. OpenID-Connect-as-a-service provides a common method to integrate with traditional and cloud-based DI systems; OpenID-Connect-as-a-service identifies and authenticates users of desktop apps and mobile apps.

We propose the design of OpenID-Connect-as-a-service (Figure 1) that provides universal identity management and ease of applying consolidated authentication mechanism, even advanced authentication technology (e.g., biometrics and hardware authentication devices), to authenticate all users (using traditional desktop or mobile devices) who request to access resources stored in traditional or cloud-based DI systems. Canada Health Infoway[4] stated that the healthcare services deployed in private or community cloud, rather than public cloud, can provide equivalent security level to traditional computing models. So deploying PACS systems and DI-r's to private cloud or community cloud is preferred cloud-based DI solution. We introduce OpenID Connect for creating an identity management and authentication ecosystem for cloud-based DI systems and for a hybrid environment constituting both traditional and cloud-based DI systems. OpenID Connect provides a simple, standard way to delegate application login to third party who continuously invests in advanced authentication techniques. As a case study, we developed a prototype for implementing an authentication service using OpenID Connect, and then integrated OpenID-Connect-as-a-service with DI services from PACS systems.

The main contributions of this paper can be summarized as follows: i) designing a user-centric decentralized authentication service for cloud-based DI systems; ii) providing a common OpenID-Connect-as-a-service API to easily integrate with DI service providers, web-based and mobile applications; and iii) utilizing "OpenID Connect" as an open source authentication delegation, and also providing user attribute claims to feed existing authorization services in the integrated systems. The remaining of this paper is organized as followings. Related work is discussed in Section 2, and the relevant background technologies are presented in Section 3. In Section 4 our OpenID-Connect-as-a-service design architecture and workflow are explained. Section 5 is allocated to the case study, and finally conclusion is presented in Section 6.

2. RELATED WORK

In this section, we discuss the approaches that are related to our work.

DICOM (Digital Imaging and Communications in Medicine) is the universal standard for PACS image storage and transmission, which defines the storage format of diagnostic images and network communication protocol.[5] WADO

(Web Access to DICOM Persistent Objects) specifies a web-based service for accessing and presenting DICOM persistent objects such as diagnostic images and diagnostic image reports.[6] WADO allows healthcare professionals simply use web browsers to obtain patient's medical information that is persisted in hospitals or image centers from anywhere. In practice, provincial DI-r is integrated with distributed PACS systems following XDS-I.b (Cross-enterprise Document Sharing for Imaging) and related IHE (Integrating the Healthcare Enterprise) integration profiles[7] for seamless sharing of DI documents among authorized users through consolidation of data in domain repositories. XDS-I.b proposed an infrastructure consisting of federated document repository (stores imaging documents) and document registry (stores imaging document metadata), which are deployed in a Service Oriented Architecture (SOA).

An image gateway that provides WADO interface has been designed and developed for DI document sharing among legacy PACS systems.[8] During the transaction of image retrieval, a document consumer can obtain an image by issuing an HTTP GET query with certain URL and some necessary parameters (e.g., study/series/object unique identifiers named "UID"). If the image can be found at PACS server, an HTTP response is sent back to the consumer including the requested image in the format of DICOM or JPEG. Then the consumer can extract the image from the HTTP response and display it on its interface such as a browser. We deploy an image gateway to each legacy PACS system, which implements web service and WADO interfaces. The image gateway accepts the access requests from web client and forwards the DICOM-compliant requests to PACS server. The architecture of XDS-I.b is based on SOA so that it is easy to deploy web service that supports XDS-I.b interface at DI-r. The protocols employed for communicating among DI systems (e.g., WADO and XDS-I.b) are not suitable to exchange data with mobile applications. A RESTful image gateway for multiple medical image repositories is presented in paper[9], where the interface is implemented as RESTful web service. Similarly, we add the RESTful interface to the image gateway to server mobile clients.

The existing open source cloud solutions have little flexibility in authentication since they are based on proprietary mechanisms. Khan et al.[10] proposed OpenID-authentication-as-a-service in the open source cloud OpenStack, and designed a decentralized authentication service in OpenStack using OpenID as an open authentication platform. Two APIs are defined: authentication API and identity verification API. Ma and Sartipi[11,12] introduced an agent-based infrastructure for secure medical image sharing between legacy PACS systems which allows for capturing PACS communication messages and authenticating users against OpenID protocol. Implementing OpenID Connect as an authentication service at the web server of image gateway is not our complete target. OpenID Connect also provides claims of attribute about the authenticated users, which can be integrated with the authorization workflow inside DI systems to make access decisions.

OpenID Connect increases the security of integrated systems by putting responsibilities for user authentication to the most expert third party service providers. The organizations that contribute to OpenID Connect are leaders in the developing of advanced authentication technologies such as bi-factor and multi-factor authentication and deploying them at OpenID Providers. In addition, the integrated systems still have options to manage their own user information and relationship but outsource the expensive, high-risk tasks of identity verification to external professional service providers. Kakizaki and Tsuji[13] proposed a decentralized user attribute information management method using OpenID Connect for identity verification. The system relies on OpenID Connect to verify identity may prefer to manage user information independently. By assigning a uniform resource identifier (URI) for all attributes, OpenID Connect identity provider only persist user's unique ID and related attribute URIs. This feature caters to the healthcare organizations that have concerns about exposing some sensitive patient information to a third party.

3. BACKGROUND

In this section, we provide an overview of different standards of Single Sign-On solutions, and discuss the best-practice of identity management and authentication frameworks in different scenarios.

3.1 Single Sign-On Solutions

SAML (Security Assertion Markup Language) is an XML-based, open-standard protocol that provides a mechanism to exchange authenticated subject information across domain boundaries, in particular, between identity provider (SAML authority) and service provider (e.g., web-based services).[14] SAML relays on an explicit trust relationship between service provider and identity provider, which means the selected service providers have to be coded in advance into service providers.

OAuth is an open protocol to allow secure authorization from web, mobile and desktop applications.[15] OAuth provides client application an access token for granting access to protected resource on behalf of the resource owner without sharing credentials such as a password.

OpenID is an open and decentralized authentication standard that provides a way to verify a user for co-operating sites (known as service providers) without sharing user credential or other sensitive information to service providers.[6] Unlike SAML, OpenID provides an identity provider discovery protocol which dynamically discovers the corresponding identity provider once a user unique ID is given. OpenID is considered obsolete by the introduction of "OpenID Connect" that is also produced by OpenID Foundation. Apart from identity verification, OpenID Connect allows service providers to use more extensible features such as encryption of identity data, dynamic discovery of identity provider, session management, and to obtain user attributes after authentication.[7]

3.2 Identity Frameworks under Different Scenarios

Figure 2 indicates four typical scenarios of DI systems and the best-practice of identity frameworks under different scenarios.

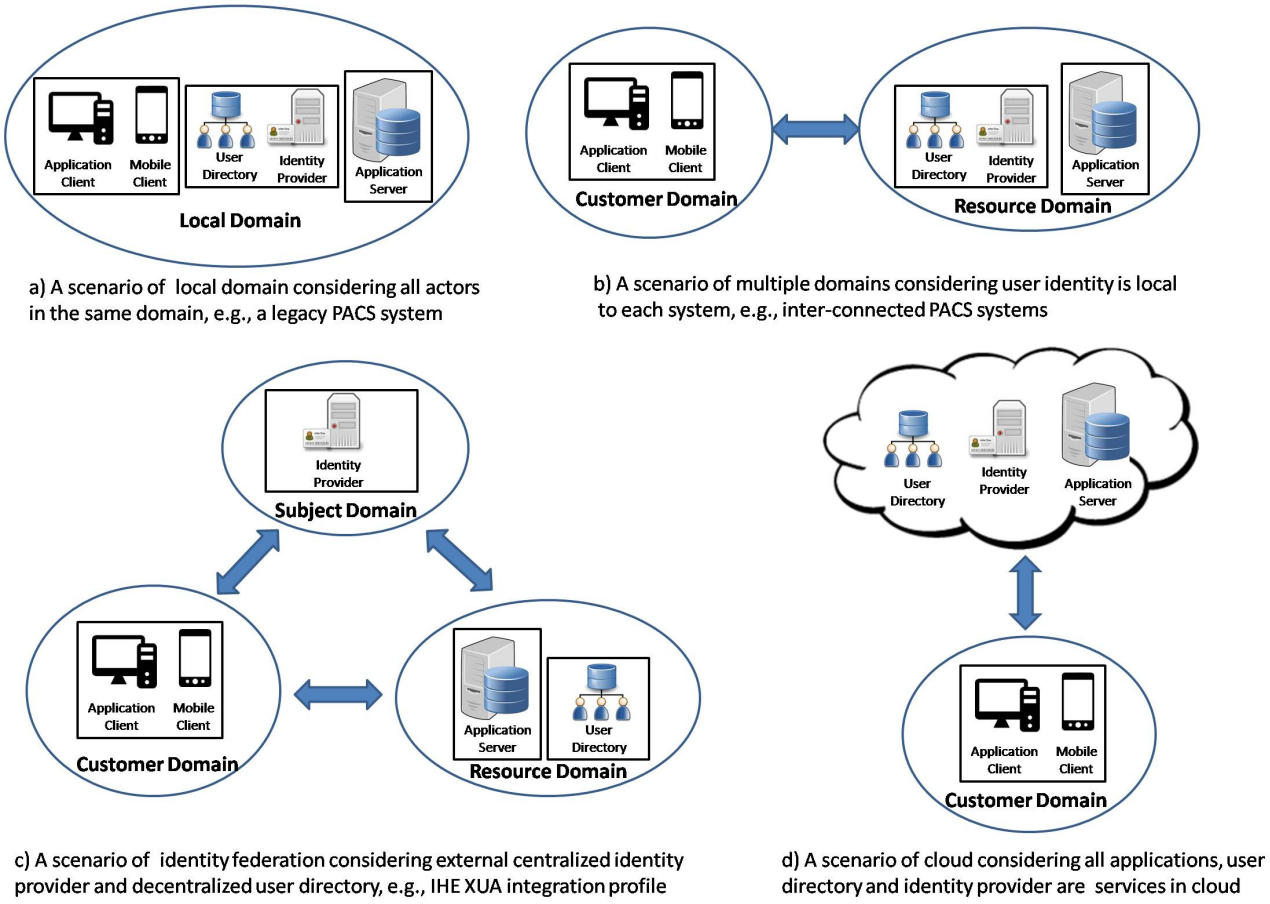

Figure 2. The best-practice of identity frameworks under different scenarios: the issuer of request can be desktop application client or mobile client; the access request affecting protected resource is located at application server; identity provider is responsible for identity verification and issuing dedicated security token; the user directory stores user attributes.

a) The simplest scenario has just one local domain where all actors involved in the authentication flow, including application clients, application server, identity provider and user directory, are residing in the same domain. This is the case for the legacy PACS system internal infrastructure.

b) With the increasing demand of DI document sharing, legacy PACS systems are interconnected across distributed environment. Each local domain is responsible for ensuring the restricted resource is adequately protected. A key challenge of this trusted model is the lack of federated capabilities. Managing and

authenticating users locally imposes a significant administrative burden to ensure that persons are uniformly identified in each system.

c) Cross-enterprise User Assertion (XUA) provides identity federation by using external centralized identity provider to generate identity claims that are communicated cross enterprise boundaries.[5] The participating enterprises that are unwilling to expose the sensitive user information to third party identity provider can have their own user directory. Typically the user identity claims relays on SAML.

d) The focus of this work is providing a common and easy identity infrastructure under a scenario that all applications, identity providers, and user directories are deployed as services in cloud environment.

4. OPENID CONNECT AS A SERVICE

OpenID Connect in cloud-based DI systems provides a decentralized user-centric authentication solution for DI services. The users have complete control over their own identity management and receive convenient SSO experience. Furthermore, OpenID Connect works with any standard Internet browser without any client-software requirement so that the users, physicians and patients, can set up their devices and applications independently to access DI documents from anywhere. Meanwhile, the DI service providers want a universal identity management and authentication method building on common platform and open standard, to reduce the IT cost. In some cases, DI service providers containing patient's sensitive information are more willing to run their own internal user information management and authorization rather than delegating a third party. The design of OpenID-Connect-as-a-service solves all these problems by providing a common integration method, and is open to integrate with decentralized user attribute management and authorization flow in existing systems.

In the architecture of integrating OpenID Connect Provider with DI services, OpenID Connect authentication is implemented as a service named "AuthN Service". Two operations are defined in AuthN Service: authentication request from DI services; and user information query from authorization service. "AuthZ Service" represents the authorization service in existing DI systems. AuthN Service provides the attributes of authenticated users which feeds AuthZ Service to make access control decisions.

Taking WADO service as an example, Figure 2 shows the sequences of user accessing DICOM image stored in PACS system from browser. The user knows the URL of WADO service and first requests to view a DICOM image over browser. The request is sent to WADO service through HTTP/HTTPS, using DICOM UIDs as query parameters. WADO service invokes an authentication operation of AuthN Service. AuthN Service redirects user's browser to a SSO page and asks for an OpenID unique identifier (simply called "OpenID identifier") that includes OpenID Provider endpoint and the associated user account to that endpoint. An OpenID identifier can be given in the form of an e-mail address or in URL syntax. For example, an OpenID identifier for a specific user may look like "myname@example.com" or "http://example.com/myname". AuthN Service can dynamically discover the OpenID Provider (simply called "OP") location using the endpoint URL http://example.com/, and obtain the OP's configuration metadata. In order to utilize OP services, AuthN Service needs to register with OP as a client, providing information about itself to OP. After registration, AuthN Service sends authentication request containing the desired request parameters to OP. OP redirects user's browser to a login page to authenticate the user. Any method to authenticate user can be used (e.g., password, credentials, information card, and biometrics). As a successful authentication response, OP returns an Authorization Code to AuthN Service. Access Token is what we need to access protected resource, and Authorization Code is a temporary credential that makes sure Access Token can be sent on secured connection. Once an Authorization Code is obtained, AuthN Service can use this code to obtain an ID Token (JSON Web Token that contains claims about the authentication event) and an Access Token (JSON Web Token that represents the responding authorization grant). Upon obtaining ID Token and Access Token, AuthN Service sends the authentication result to WADO service. At this point, the authentication process is completed. WADO service invokes AuthZ Service to obtain an authorization decision. AuthZ Service invokes a user information query operation of AuthN Service. Using Access Token obtained through authentication flow, AuthN Service makes a request to OP for claims (normally represented by a JSON object that contains a collection of name and value pairs) of the authenticated user such as user's full name, email address, health card number, and picture. Such user attributes feed AuthZ Service to make authorization decisions. Once the access request is granted by AuthZ Service, the DICOM object can be viewed in the browser.

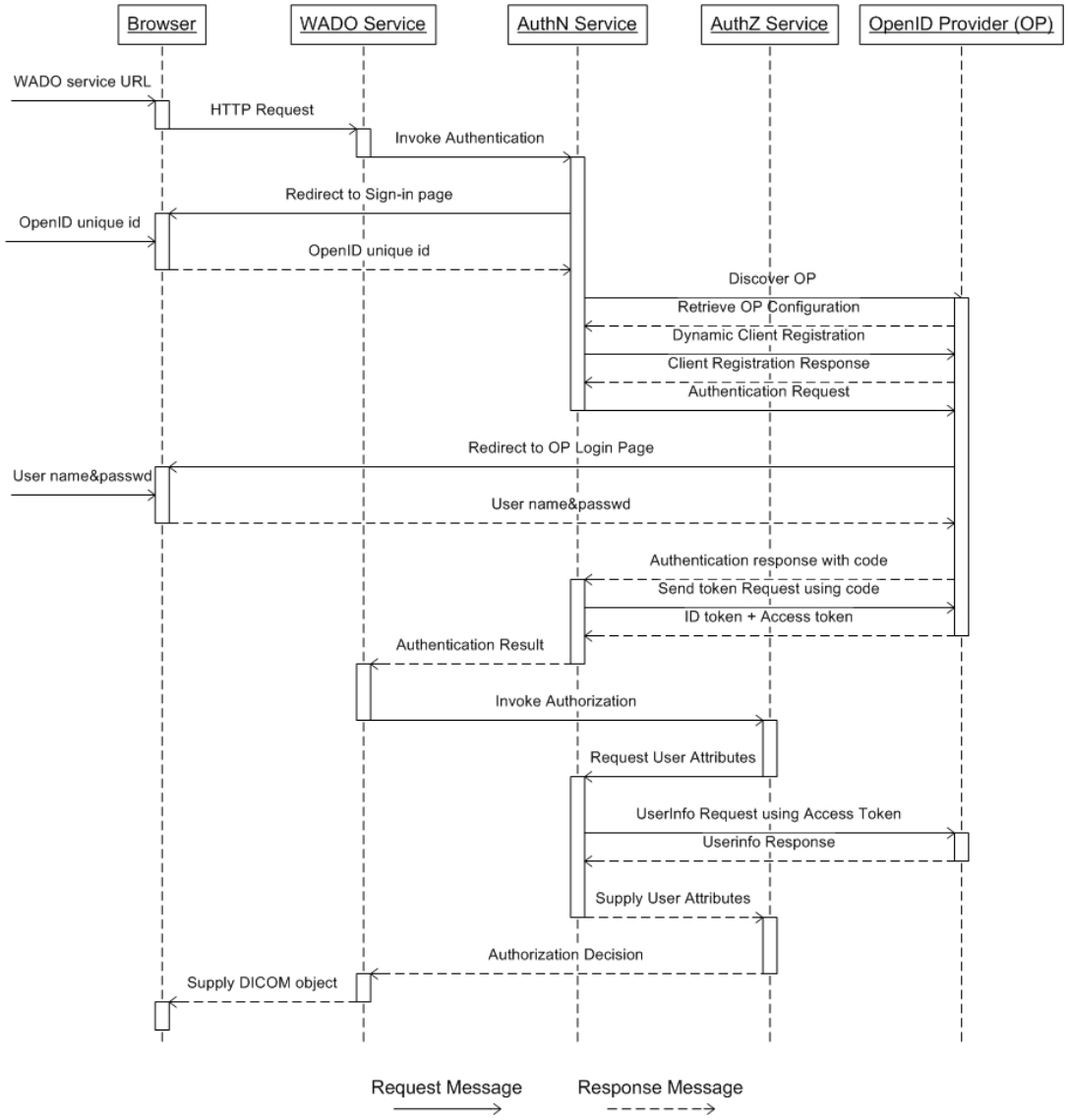

Figure 3. The sequence for authentication workflow using OpenID-Connect-as-a-services when user issues a request from browser to access a DICOM object.

AuthN Service is in charge of interactions with OpenID Provider, and managing (obtaining, verification and refreshing) ID Tokens and Access Tokens. OpenID-Connect-as-a-service design separates authentication from authorization, and provides common and easy APIs for integration. The existing DI services only need to call AuthN Service API to do authentication operation, and existing AuthZ Service in DI systems need to call AuthN Service API to retrieve user related information.

5. PROTOTYPE IMPLEMENTATION

5.1 Implementing OpenID-Connect-as-a-Service

OpenID Connect uses REST/JSON message flows which are easy for developers to integrate, compared to preceding federated identity protocols. Open source tool "PYOIDC" is a complete implementation of OpenID Connect specifications written in Python.[16] It is developed and maintained by members of OpenID community. Open source tool CherryWado is selected to simulate a WADO server which is also written in Python.[17] CherryWado uses Web Server Gateway Interface (WSGI) server to implement Python web applications. DICOM images are handled by Python

Imaging Library (PIL) that provides powerful image processing and graphics capabilities. The WADO service does not make any assumption about how/where the DICOM files are stored. We implemented a DICOM access layer that specifies a way to retrieve the corresponding DICOM files from DICOM image repositories. Figure 4 indicates the class diagram of our prototype implementation.

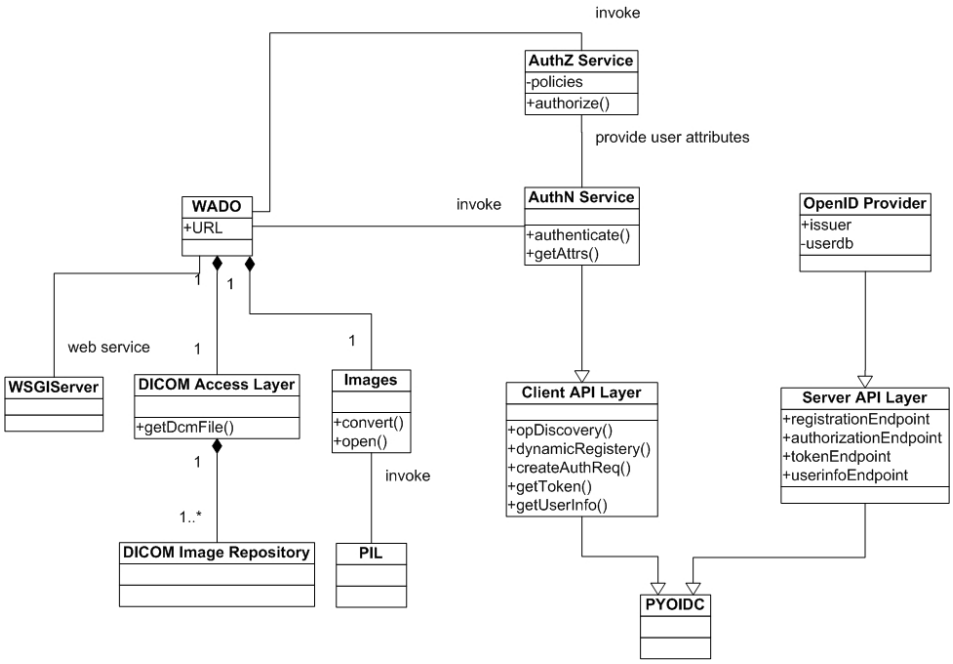

Figure 4. The class diagram describes the structure of prototype implementation.

- ***WADO***. It provides a mechanism for accessing DICOM objects (e.g., medical images, and medical image reports) through HTTP/HTTPS protocols using DICOM UIDs as query parameters. By using Python web application framework WSGI, the access requests are routed to WADO service based on the target URL. Once an access request arrives, WADO first invokes the method authenticate() of AuthN Service to identify and verify the user, then invokes the method authorize() of AuthZ Service to make authorization decision if the user is authenticated successfully. After obtaining grant, WADO retrieves DICOM file from DICOM image repository using "DICOM Access Layer" as an intermediary. At last the DICOM file is converted to more compatible format such as JPEG through class Image and is displayed in browser.
- ***AuthN Service.*** It exposes an API authenticate() to WADO service to do authentication, and an API getAttrs() to AuthZ Service to retrieve user related attributes. To set up an OpenID Connect Relaying Party (RP, which is service provider requiring user authentication and claims from OpenID Provider), a Client API Layer is implemented that inherits the functionalities provided by PYOIDC. The method opDiscovery() is used to find the location of desired OpenID Provider, and dynamicRegistery() registers RP to OpenID Provider as a client on behalf of AuthN Service. The methods createAuthReq(), getToken(), and getUserInfo() implement the interactions with OpenID Provider following the process presented in Section 4.
- **OpenID Provider.** Through a new layer Server API Layer, an OpenID Provider is implemented using the functionalities provided by PYOIDC. The class members RegistrationEndpoint allows a client to register at OpenID Provider; authorizationEndpoint is in charge of authenticating users and issues Authorization Code to client; tokenEndpoint issues ID Token and Access Token for authenticated users; and userinfoEndpoint returns attributes about the authenticated user when Access Token is presented. The class member of OpenID Provider "issuer" enables determining the location of the OpenID Provider; a userdb that persists and manages user information, centralized or decentralized, can be integrated with OpenID Provider.

- **AuthZ Service.** A typical authorization service makes determination for an access request based on available information (e.g., user attributes, protected resource attributes) and applicable security policies. The user related information can be retrieved from AuthN Service from OpenID Provider.

5.2 Experimental Result

Following is the experiment result of the implementation prototype presented in subsection 5.1. A user account is predefined in OpenID Provider containing: OpenID identifier "weina@example.com", username and password, and user attributes such as address and phone number. An access control policy is created at AuthZ Service: only users living in a specific community can access images from a specific DICOM repository.

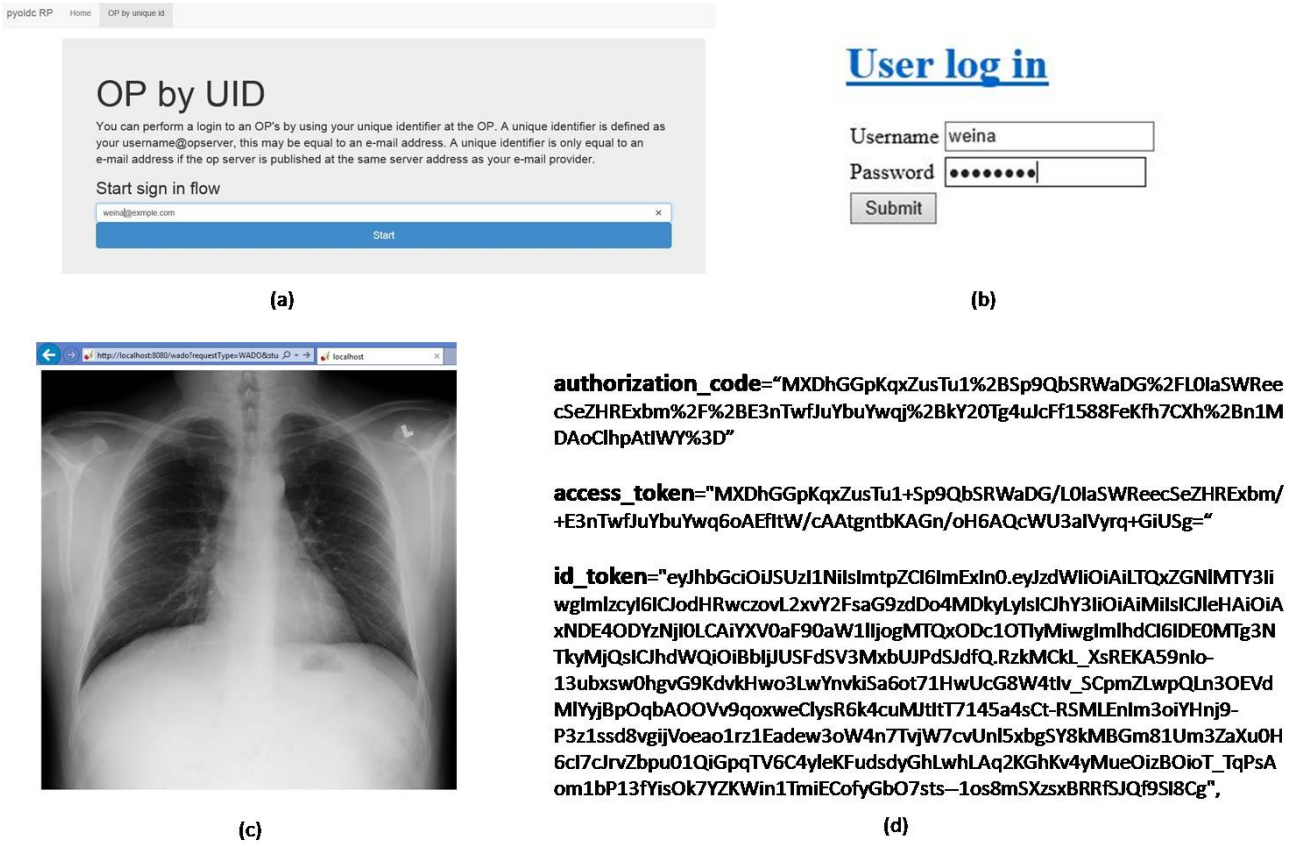

Figure 5. (a) Ask user to input an OpenID unique identifier. A unique identifier is defined as username@opserver. "example.com" specifies the OpenID Provider server we implemented. (b) User login page. (c) The diagnostic image is displayed in the browser. (d) A snapshot of Authorization Code, Access Token and ID Token exchanged between AuthN Service and OpenID Provider.

Firstly, the user wants to view an image stored in DICOM repository through browser in the format of JPEG. She enters the WADO service URL, DICOM study, series and object UIDs, and content format in the browser that may look like http://localhost:8080/WADO?requestType=WADO&studyUID="1.3.76.13.10010.0.5.74.3996.1224256625.4053"&seriesUID="1.3.12.2.1107.5.4.4.1053.30000008100608242373400002493"&objectUID="1.3.12.2.1107.5.4.4.1053.30000008100608324685900001822"&contentType=image/jpeg".

WADO service receives the access request and asks for AuthN Service to do authentication. Figure 5-(a) shows redirected page asking for user to enter OpenID identifier. An email address is entered which includes the unique account name "weina" and OpenID Provider host "example.com". AuthN Service is able to find the location of OpenID Provider using "example.com". OpenID Provider redirects to user login page and needs user input username and password as shown in Figure 5-(b). After authentication and authorization are completed, the DICOM image is retrieved from DICOM file repository and displayed in browser in the format of JPEG, Figure 5-(c). The Authorization Code, Access Token and ID Token, exchanged between AuthN Service and OpenID Provider, are shown in Figure 5-(d).

6. CONCLUSIONS

In this paper, we introduced OpenID-Connect-as-a- service to provide user-centric Single Sign-on solution in the cloud-based diagnostic imaging systems. It allows the user to use one account to sign in to multiple healthcare services, without exposing password to all these services. OpenID Connect is a decentralized open authentication protocol, and provides REST based APIs that allows different types of healthcare services, including web-based and mobile applications, to delegate OpenID Provider to do authentication. The design of AuthN Service is flexible, scalable, ease of integration, and provides several of options to integrated systems. In the case of the healthcare organizations with concerns about exploring some sensitive patient information to a third party, these organizations can manage their own user information but just outsource the identity verification to OpenID Provider. Beside of user verification, our proposed AuthN Service is also able to provide user attribute claims to feed existing authorization services in the integrated systems. Moreover, OpenID Connect is open to use any modern authentication technology such as smart card and biometrics, which offers the healthcare service providers easier and faster access to the advanced identity management with lower investment. This research attempts to provide a design for common authentication services in cloud-based DI ecosystem and the implemented prototype proves the feasibility of the design.

Acknowledgement. This research was conducted with collaboration of Dr. David Koff and Dr. Peter Bak at MIIRC@M Centre of McMaster University. This research was funded by an ORF grant for the project ”Secure Intelligent Content Delivery System for Timely Delivery of Large Data Sets in a Regional/National Electronic Health Record”.

REFERENCES

[1] Branstetter, B. F. (Ed.), “Practical imaging informatics: foundations and applications for PACS professionals”, Springer, 33-47 (2009).
[2] Gauvin, A., “Status of Diagnostic Imaging Repository (DI-r) projects across Canada”, http://www.camrt.ca/ (2010)
[3] OpenID Foundation websit, http://openid.net/foundation/
[4] Canada Health Infoway, “Cloud Computing in Health White Paper”, https://www.infoway-inforoute.ca/ (2012).
[5] Onken, M., Eichelberg, M., Riesmeier, J., and Jensch, P. “Digital Imaging and Communications in Medicine,” Biomedical Image Processing. Springer Berlin Heidelberg. (2011)
[6] Koutelakis, G. V. and Lymperopoulos, D. K., "PACS through web compatible with DICOM standard and WADO service: advantages and implementation," In Engineering in Medicine and Biology Society, EMBS'06. 28th Annual International Conference of the IEEE, 2601-2605 (2006).
[7] “IHE Radiology Technical Framework, Volume 1: Integration Profiles”, http://www.ihe.net/ ,190-205 (2012).
[8] Simalango, M. F., Kim, Y., Seo, Y. T., Choi, Y. H., and Cho, Y. K., “XDS-I Gateway Development for HIE Connectivity with Legacy PACS at Gil Hospital,” Healthcare informatics research, 19(4), 293-300 (2013).
[9] Valente, F., Viana-Ferreira, C., Costa, C., and Oliveira, J. L., “A RESTful image gateway for multiple medical image repositories,” Information Technology in Biomedicine, IEEE Transactions on, 16(3), 356-364 (2012).
[10] Khan, R. H., Ylitalo, J., and Ahmed, A. S., “OpenID authentication as a service in OpenStack,” In Information Assurance and Security (IAS), 7th International Conference, IEEE, 372-377 (2011).
[11] Ma W, and Sartipi K., “An Agent-Based Infrastructure for Secure Medical Imaging System Integration," Computer-Based Medical Systems (CBMS), IEEE 27th International Symposium, 72-77 (2014)
[12] Sartipi, K.,Kuriakose, K., and Ma, W., “An Infrastructure for Secure Sharing of Medical Images between PACS and EHR Systems,” International Conference on Computer Science and Software Engineering (IBM CASCON), 245-259 (2013)
[13] Kakizaki, Y., and Tsuji, H., “A Decentralized Attribute Management Method and its Implementation,” International Journal of Information Processing & Management 3.1 (2012)
[14] SAML 2.0 wiki, http://en.wikipedia.org/wiki/SAML_2.0
[15] OAuth websit, http://oauth.net/
[16] Source code of OpenID Connect implementation in Python, https://github.com/rohe/pyoidc
[17] Source code of CherryWado, https://github.com/malaterre/GDCM/tree/master/Applications/Python