

# ***OpenHRE Security Architecture***

***(DRAFT v0.5)***

# Table of Contents

- Introduction** -----2
- Assumptions** -----2
- OpenHRE Architecture**-----3
- Technologies**-----4
  - SASL-----4
  - Kerberos -----4
  - SAML -----4
- Security in OpenHRE**-----5
  - Access Control and Policy Management in OpenHRE -----6
    - Role-Based Access Control (RBAC) -----6
    - Instance-Based Access Control -----6
    - Implementing access control and policy management -----6
  - User Authentication and Authorization in OpenHRE -----8
    - Within a RHIO -----9
    - Inter-RHIO ----- 10
- References**----- 11

## Introduction

The following statement is from the final rule of Health Insurance Reform: Security Standards issued by Department of Health and Human Services [1]:

“The Department of Health and Human Services (HHS) Medicare Program, other Federal agencies operating health plans or providing health care, State Medicaid agencies, private health plans, health care providers, and health care clearinghouses must assure their customers (for example, patients, insured individuals, providers, and health plans) that the integrity, confidentiality, and availability of electronic protected health information they collect, maintain, use, or transmit is protected. The confidentiality of health information is threatened not only by the risk of improper access to stored information, but also by the risk of interception during electronic transmission of the information. The purpose of this final rule is to adopt national standards for safeguards to protect the confidentiality, integrity, and availability of electronic protected health information. Currently, no standard measures exist in the health care industry that address all aspects of the security of electronic health information while it is being stored or during the exchange of that information between entities.”

Confidentiality of medical records is of major concern in the design of the OpenHRE system. To safeguard patient privacy, the OpenHRE product will enforce strict access control policies determining who can see what data under what circumstances, and what happens when these rules are not followed. The rules will comply with HIPAA and the more stringent California Medi-Cal (Medicaid) regulations. The policies are to be embedded in the OpenHRE product technology so that users are forced into compliance with the access control policies. These policies include the rules for authentication, informed consent, data holder over-rides, and other elements such as logging, screen locking and auditing detailed herein. In general, the policies presume access to data when the established rules are met: any practitioner who can electronically identify him or herself as an authorized participant, has patient consent, and has not had his or her access over-ridden by a data holder, can get access to the requested patient's data.

As regulations and policies regarding confidentiality and security of patient medical records continues to evolve, the embedded rules and procedures will also change, and so the product will incorporate flexibility in authentication and record provision at a number of levels, including but not limited to provider access, patient access, data stream security, clinic level data security and integrity, and the right of the patient, clinic, and system to block access to some or all records, and the right of the patient, clinic and system to audit record access trails.

In order to achieve the above goals, an efficient security architecture is needed for OpenHRE. The remainder of the text provides a simplified overview of some of the proposals for such a security architecture.

## Assumptions

The following assumptions shape the OpenHRE Security Architecture:

1. Patient data, including demographics and health records, are under the control of the clinic, office, or hospital that originates the data, even if some of the data is physically housed or replicated at a central site.

2. Clinical data will not be housed at a central site, although it may be temporarily cached there.
3. User authorization and authentication is under the control of the individual clinics, offices, and hospitals.

## OpenHRE Architecture

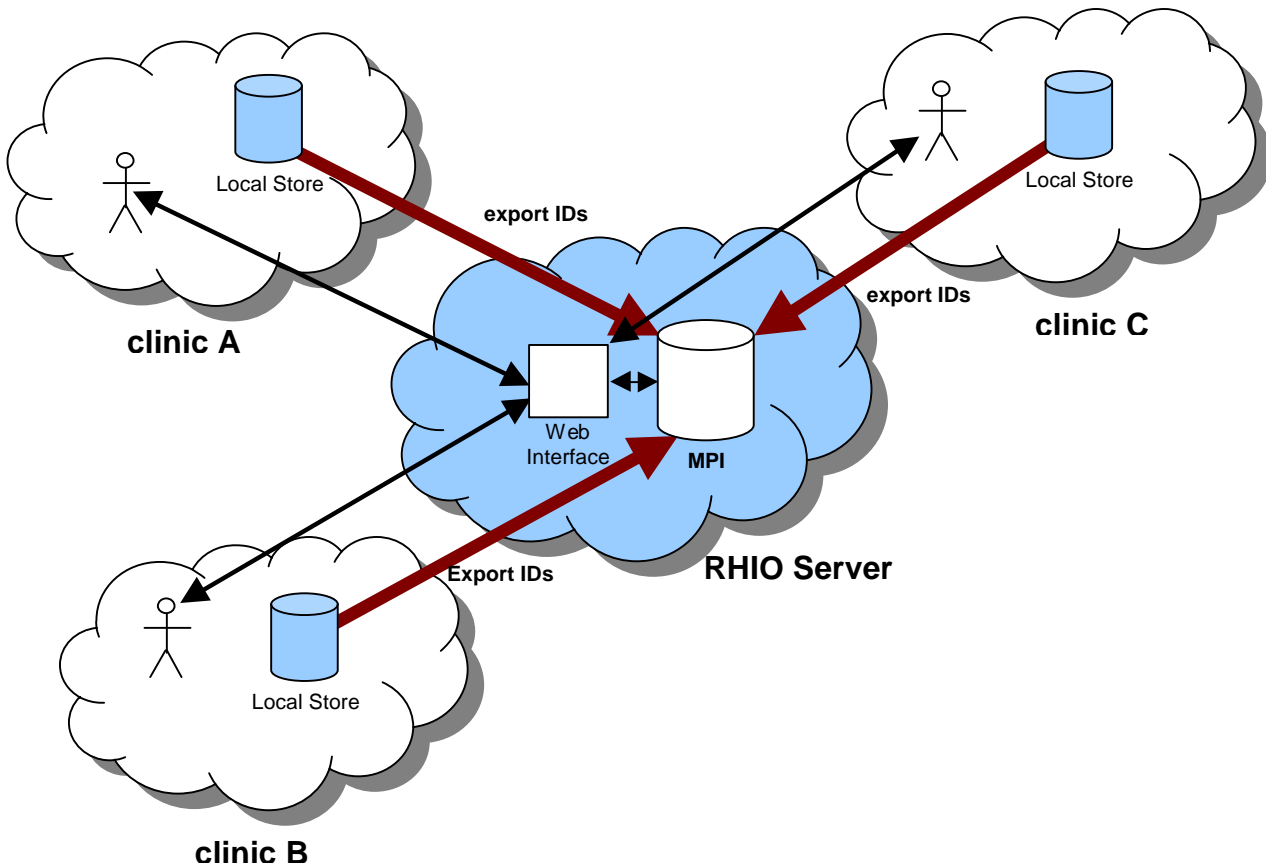


Figure 1: RHIO in OpenHRE

The above figure depicts a generic view of a Regional Health Information Organization (RHIO) using the current OpenHRE architecture. A clinic within a RHIO has its own domain (a set of unique patient identifiers, identity/demographic data, clinical records, users, and roles) and exports its patient demographic information to the RHIO *Master Patient Index (MPI)*. The users within a clinic's domain can access this information from the RHIO's MPI via a web based interface. The RHIO Server has the ability to access clinical data within each clinic's domain and consolidate it for presentation to authorized users.

RHIOs can also be arranged in a tree-like structure, where an upper-level parent RHIO's MPI not only includes the patient demographic information from the clinics within its own realm, but also from its child RHIOs.

## Technologies

The following are some of the authentication and authorization technologies that can be used to create an efficient architecture for OpenHRE security.

### SASL

SASL (Simple Authentication and Security Layer) [2] is a method for adding authentication support to connection-based protocols. To use SASL, a protocol includes a command for identifying and authenticating a user to a server and for optionally negotiating protection of subsequent protocol interactions. If its use is negotiated, a security layer is inserted between the protocol and the connection. In short, SASL is a framework, which supports many *pluggable* user authentication mechanisms, such as plain passwords, DIGEST MD5 and Kerberos v5.

### Kerberos

Kerberos [3] is a network authentication protocol, which is designed to provide strong authentication for client/server applications across insecure network connections by using secret-key cryptography. It allows entities communicating over networks to prove their identity to each other while preventing eavesdropping or replay attacks. It also provides for data stream integrity (detection of modification) and secrecy (preventing unauthorized reading) using cryptographic ciphers such as DES.

Kerberos works by providing principals (users or services) with tickets that they can use to identify themselves to other principals and secret cryptographic keys for secure communication with other principals. A ticket is a sequence of a few hundred bytes. The ticket can then be embedded in virtually any other network protocol, thereby allowing the processes implementing that protocol to be sure about the identity of the principals involved.

Kerberos provides for mutual authentication and secure communication between principals on an open network by manufacturing secret keys for any requestor and providing a mechanism for these secret keys to be safely propagated through the network. Kerberos does not, strictly speaking, provide for authorization or accounting, although applications may use their secret keys to perform those functions securely.

OpenHRE can use Kerberos over SASL for user authentication via the local domain, when access is requested.

### SAML

The Security Assertion Markup Language (SAML) [4] is an XML-based framework for exchanging security information. This security information is expressed in the form of assertions about subjects, where a subject is an entity (either human or computer) that has an identity in some security domain. A

typical example of a subject is a person, identified by his or her email address in a particular Internet DNS domain. Assertions can convey information about authentication acts performed by subjects, attributes of subjects, and authorization decisions about whether subjects are allowed to access certain resources. Assertions are represented as XML constructs and have a nested structure, whereby a single assertion might contain several different internal statements about authentication, authorization, and attributes. Note that assertions containing authentication statements merely describe acts of authentication that happened previously. Assertions are issued by SAML authorities, namely, authentication authorities, attribute authorities, and policy decision points. SAML defines a protocol by which clients can request assertions from SAML authorities and get a response from them. In the case of OpenHRE, a SAML client is the RHIO's *Authentication and Authorization (AA)* server and the AA server in the user's domain is the SAML authority (more details will be discussed later).

SAML specifies XML constructs for requests and responses (including assertions), which make use of XML signatures to provide data integrity and authentication. An XML Signature is an XML syntax used for representing digital signatures on digital content and there are specific procedures for computing and verifying such signatures. XML signatures are usually based on X.509 certificates possessed by the signer.

Adopting SAML not only solves the problem of efficiently exchanging security information between servers, but can also help in providing single sign-on (SSO) capabilities once the need arises to enable a user to view patient information from different RHIOs. Inter-RHIO details are not discussed in this document.

## Security in OpenHRE

From the architectural point of view, a RHIO is comprised of a RHIO central server (which includes a web server, an MPI, and an *Authentication and Authorization (AA)* server), and servers installed at the clinics within the RHIO (which include a user directory server and an AA server), and its users. As explained earlier, a RHIO can also act as a parent for other RHIOs, enabling inter-RHIO collaboration.

A clinic's user directory could also be housed in a partition of a user directory located centrally in the RHIO. The key point is that the clinic's user directory is administered and maintained by the clinic.

All the servers, including the RHIO central server and servers installed at individual clinics, need to install X.509 public-key certificates issued by a trusted authority. Who the trusted authority should be and how the certificates are to be installed are administrative decisions and are out-of-scope of this document.

All communication between a user's browser and a web server should be secured by SSL/TLS using the server's certificate. All communication between the servers installed at different locations, including the RHIO central server, must be authenticated and protected using the public-key infrastructure using their X.509 certificates.

All the users belonging to a particular clinic should have an account maintained by a locally installed directory server based on LDAP. Locally maintaining user accounts allow each clinic to manage its users more efficiently as compared to informing a central server each time a user needs to be added, modified or deleted. The directory server also maintains attributes, such as roles, for each user. Each

clinic has an AA server, which can be contacted to authenticate a local user and to specify user roles for access decisions.

## Access Control and Policy Management in OpenHRE

The users in OpenHRE have different job functions, which mandate granting different levels of access to patient information. The following discusses what types of restrictions can be applied and how OpenHRE will implement them.

### **Role-Based Access Control (RBAC)**

Users within a system can be restricted access to resources according to their respective roles within the organization using RBAC. RBAC consists of three entities: *user*, *role* and *permission*, where user and role & role and permissions are many-to-many relationships, i.e., users are assigned roles and roles have permissions for the resources.

RBAC facilitates *role hierarchy*, for example, a physician can inherit permissions assigned for doctors; and *separation of duties*, for example, a bank teller cannot be a customer of the same bank at the same instance of time.

### **Instance-Based Access Control**

Providing even finer granularity for access control can be done using instance-based access control, which assigns permissions to individuals. For example, only doctor A can edit patient A's records. This can be extended to provide grouping of instances, for example, only doctor A, B & C can edit patient A's records. Access control of this kind is suitable for the healthcare industry, since a patient often has a single doctor (or a group of doctors) responsible for his or her treatment.

### **Implementing access control and policy management**

Access control is enforced in a system based on policies and rules within its domain. After authenticating the user, the system reads these rules to determine what permissions are granted to the user and restricts the access accordingly.

There should be a common set of roles defined within a RHIO, so that all clinics can grant a uniform set of roles to its users. For example, this means that a role called "physician" should be consistent within a RHIO's domain.

There are two ways of managing policies and rules. One way is to maintain policies at individual clinics and each time when a user logs into a RHIO server to access the patient demographic information, the server requests all the applicable clinics to authorize the user. Though this method gives more control to individual clinics, it increases the amount of information that needs to be exchanged when a user needs to be authorized. An alternate is to reduce the amount of communication at runtime and

propagate policy and rule changes up to a central server, which would be responsible to authorize the user.

If the second approach is adopted, access decisions are made using a policy file or database at the RHIO central server, which is a collection of rules from different clinics regarding accessing their patient information. These rules are based on the level of trust between clinics. For example, clinic A might have a rule that no physician except the ones from clinic A and B can edit the patient information from clinic A. These rules are propagated from a clinic to the RHIO central server's policy file or database whenever they are updated through an out-of-band communication (as shown in the following figure). The same approach is applicable in case of inter-RHIO information access.

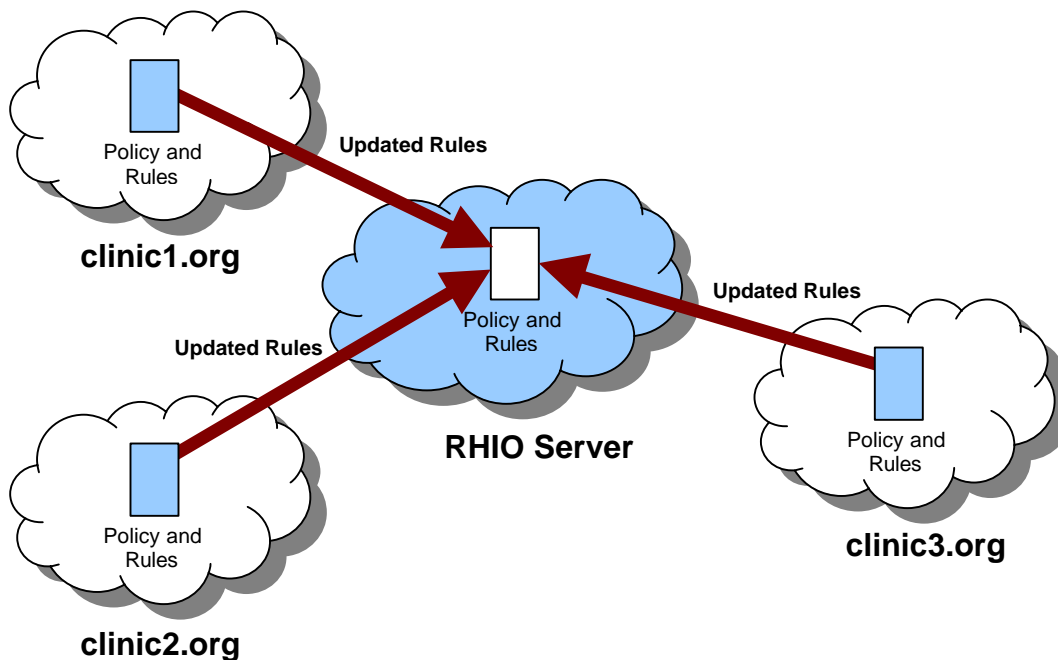


Figure 2: Policy information exchange in a RHIO

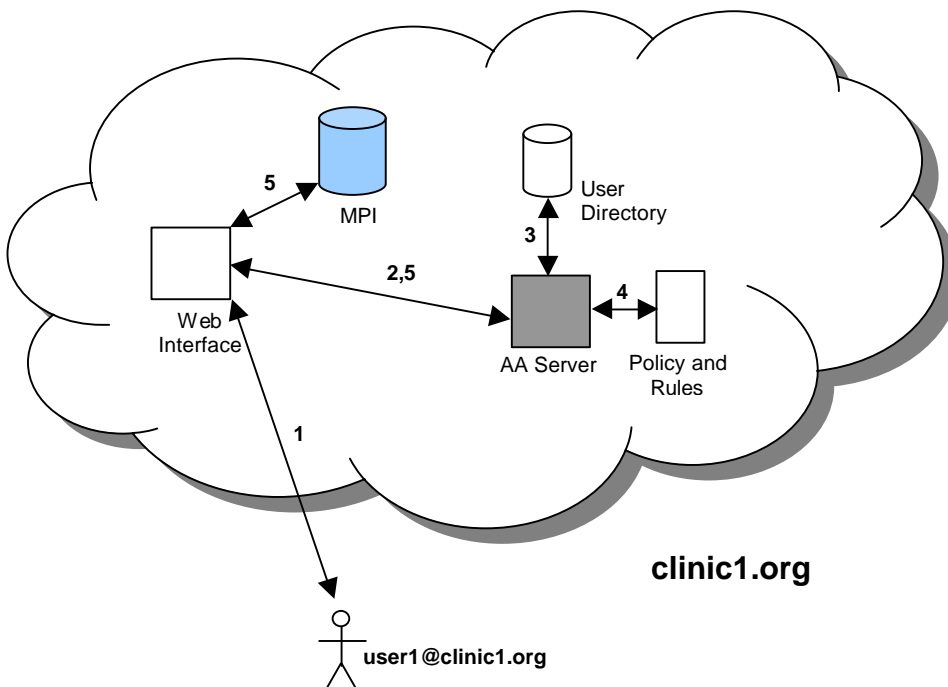
Access decisions can be made using an extended version of OpenEMed's Resource Access Decision (RAD) framework (which supports instance-based access control) or an XACML (eXtensible Access Control Markup Language) [5] based policy management framework. The details will be discussed in a later version of this document.

## User Authentication and Authorization in OpenHRE

In this section, we will discuss three possible scenarios where user authentication and authorization are required.

### Within a clinic's own domain

The following scenario occurs when a user accesses the MPI within the clinic's domain. This assumes that the clinic has its own MPI, and the user only wishes to access information within the clinic.



**Figure 3: User Authentication and Authorization within a clinic's own domain**

1. The user requests to login via the web interface of the clinic's server to access the clinic's MPI.
2. The web interface calls the AA (*Authentication and Authorization*) server within the clinic's domain to authenticate him.
3. The AA server gets the user's credentials via the web interface, verifies them (using Kerberos over SASL) and fetches the user's role(s) from the local user directory.
4. The AA server then checks its local policy file or database to determine what information in the MPI should be accessible to the user, and what level of access should be granted. It then grants appropriate privileges to the user.
5. Control is returned to the web interface, which allows access to the appropriate information using a filter based on the user's privileges.

## Within a RHIO

In the following scenario, a user is authenticated and authorized before accessing the MPI within the RHIO he/she belongs to. This may happen because the clinic does not have its own MPI, or possibly because the user wishes to access records from several clinics within the RHIO. Note also that while the clinic has its own user directory, maintained by the clinic, this directory might be physically co-located with the RHIO Server.

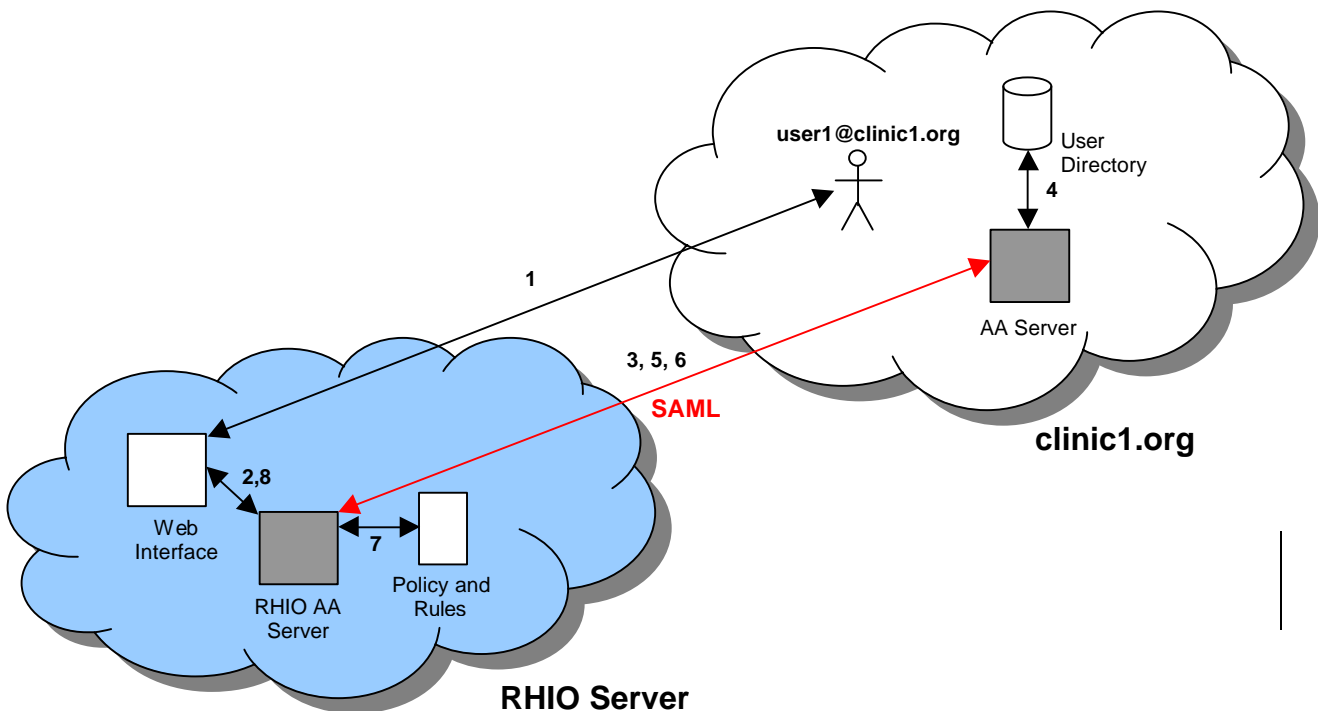


Figure 4: User Authentication and Authorization within a RHIO

1. The user requests to login via the web interface of the RHIO Server to access its MPI.
2. The web interface calls the RHIO AA server to determine the user's domain from his username (in the above diagram, the RHIO AA server determines that *user1@clinic1.org* belongs to the domain *clinic1.org*).
3. Using SAML, the RHIO AA server requests the AA server at the user's domain to authenticate the user.
4. The AA server at the user's domain gets the user's credentials from the SAML request, verifies them and fetches the user's role(s) from the local user directory.
5. The AA server at the user's domain creates a SAML assertion, which includes the authentication and attribute statements specifying the user's role(s), signs it and stores it. It then creates an artifact (which specifies the location of the SAML assertion) and returns it to the RHIO AA server.
6. The RHIO AA server sends a SAML request with the artifact to the AA server at the user's domain to retrieve the SAML assertion. The AA server at the user's domain sends a SAML response containing the SAML assertion for the user to the RHIO AA server.
7. The RHIO AA server reads the SAML assertion, checks its local policy file (which includes the policies of all the clinics within the RHIO) to determine what information in the MPI should be

accessible to the user and what level of access should be granted. It then grants appropriate privileges to the user.

- The RHIO AA server creates its own SAML assertion along with an artifact for the user for further authentication of the user. Control is returned to the RHIO's web interface, which allows access to the appropriate information using a filter based on the user's privileges.

### Inter-RHIO

In the following scenario, a user is authenticated and authorized before accessing the MPI within a RHIO (which we will refer to as the parent RHIO) above the RHIO he/she belongs to. In the following figure, both RHIO-1 and RHIO-2 are a child of the parent RHIO, who's MPI includes patient demographic information from both of them.

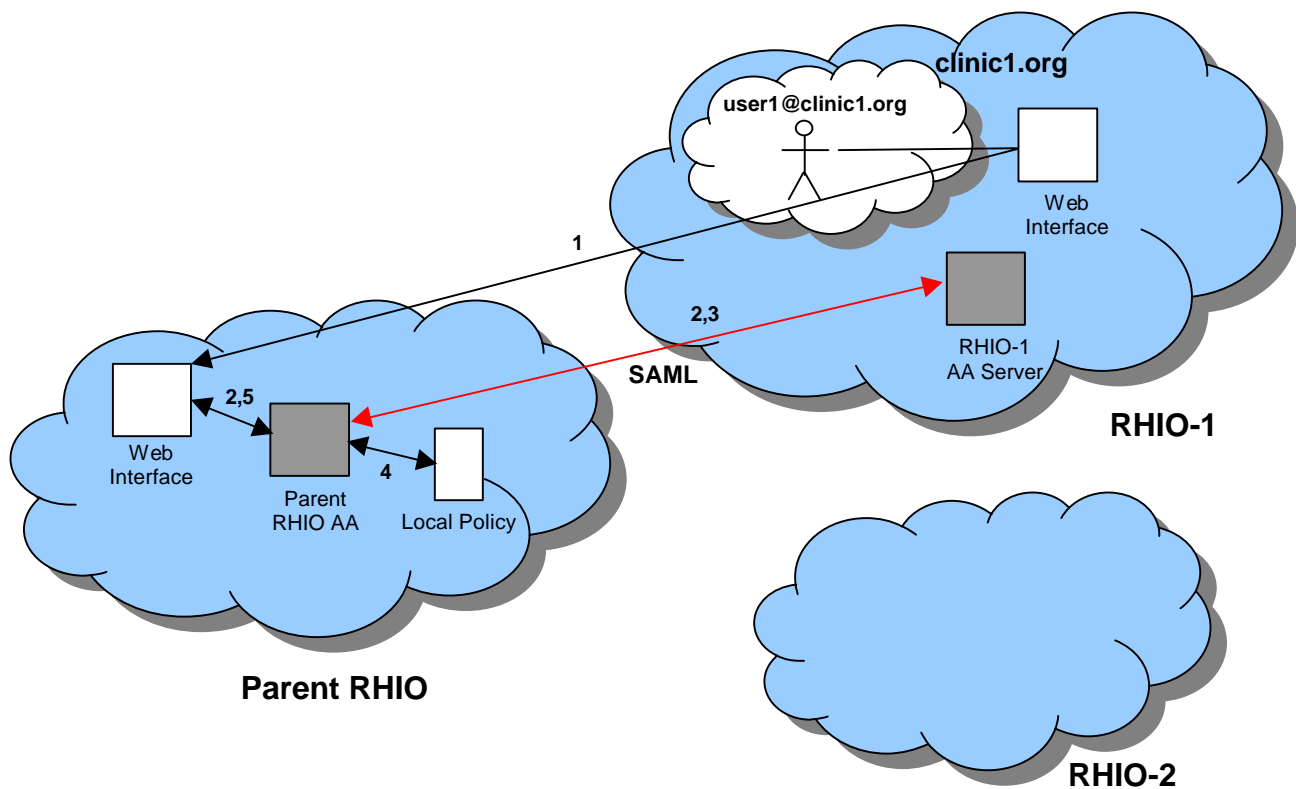


Figure 5: Inter-RHIO User Authentication and Authorization

- The user requests to access the MPI belonging to the parent RHIO, via his RHIO's (i.e., RHIO-1's) web interface. RHIO-1's web interface redirects the user to the parent RHIO's web interface along with the artifact it created for the user after authenticating him/her.
- The parent RHIO's web interface calls the parent RHIO AA server, which sends a SAML request with the artifact to the RHIO-1 AA server to retrieve the SAML assertion.
- The RHIO-1 AA server sends a SAML response containing the SAML assertion for the user to the parent RHIO AA server.
- The parent RHIO AA server reads the SAML assertion, checks its local policy file (which includes the policies from both RHIO-1 and RHIO-2) to determine what information in the MPI should be

accessible to the user and what level of access should be granted. It then grants appropriate privileges to the user. Thus, the SAML assertion enables the user to automatically sign-on to the parent RHIO's server.

5. The parent RHIO AA server creates its own SAML assertion along with an artifact for the user for further authentication of the user. Control is returned to the parent RHIO's web interface, which allows access to the appropriate information using a filter based on the user's privileges.

## References

[1] "Health Insurance Reform: Security Standards", Department of Health and Human Services. [http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=2003\\_register&docid=fr20fe03-4](http://frwebgate.access.gpo.gov/cgi-bin/getdoc.cgi?dbname=2003_register&docid=fr20fe03-4)

[2] "RFC 2222: Simple Authentication and Security Layer (SASL)", <http://www.ietf.org/rfc/rfc2222.txt>

[3] "Kerberos: The Network Authentication Protocol", <http://web.mit.edu/kerberos/www/>

[4] "OASIS Security Services TC", [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=security](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security)

[5] "OASIS eXtensible Access Control Markup Language TC", [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=xacml](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml)