Handling Authentication and Authorization

Because security is an ongoing concern and best security practices are continually evolving, an Ed-Fi REST API implementation *should not* preclude any future security enhancements. Implementations *should* reflect the current best practices where they affect the actual API structure or use. Additional layers of security *may* be employed on top of those identified here.

In order to secure a data repository, the repository *must* know who is requesting an operation, and what the requestor is allowed to do. The question of who is known as authentication; the what is authorization.

Authentication

The process of authentication provides the means of reliably identifying applications and users to an Ed-Fi REST API implementation. Both applications and users require authentication in order to maintain secure data. The OAuth 2 specification[10] is used by an Ed-Fi REST API for authentication. This specification is broad enough to handle both private and public identity providers. An identity provider issues identification information for all providers desiring interaction with the system. An authentication module verifies a security token as an alternative to explicitly authenticating the user.

Both application authentication and user authentication *should* be used. When applications do not have user authentication associated with API calls, the API *must* allow application-only authentication. When user credentials are available, the API *must* require both application and user credentials. These situations are described below.

Application Authentication

In an Ed-Fi REST API, an application is identified when it provides credentials and receives a token that is subsequently used for each interaction during a given period of time (often called a session). An Ed-Fi REST API *should* provide a method of application authentication. Two-legged OAuth using the client credentials grant flow is the *recommended* implementation for application authentication.

Applications accessing an Ed-Fi REST API *should* be fully vetted to perform as expected before production application credentials are issued.

User Claims

A claim is a piece of information about a user provided by an Security Token Services (STS)[11] to an application. Typically, this claim establishes the identity of the user. Examples of claims include: credential (login name), credential originator, email address, display name, and identification key(s) or number(s). Other times, a claim may resemble a role; it is not uncommon to see “job title,” “department,” or “organization” claims.

Claims from the STS *should* be used to identify a user. Claims *may* also be used (directly or indirectly) to provide authorization clues to an Ed-Fi REST API. User claims *should* be transmitted to the API using either an opaque “bearer” token (e.g. a GUID) where the token value is used by the server to load the user’s permissions or to encode the claims within a signed JWT that the server then verifies upon receipt.

System Applications

Some classes of applications do not naturally operate in the context of a user; they do not have user authentication associated with API calls. Examples of this type of application include: bulk load and extract tools, enterprise-wide SISs, and some classes of reporting tools. In these cases, extreme discretion should be used when considering eliminating user authentication from the security framework for these implementation scenarios.

In these cases it becomes necessary for the API to allow application-only authentication or, preferably, to operate as a “system” user with global privileges.

Authorization

Authorization is a set of mechanisms for identifying what operations can be performed, and upon which resources. Due to privacy concerns and FERPA (the Family Educational Rights and Privacy Act) regulations, it is critical that an Ed-Fi REST API implementation correctly authorizes all resource requests.

The principle of least privilege *should be* used for an Ed-Fi REST API implementation. Least privilege means that default application and user permissions are no privileges and that all privileges be explicitly granted.

Privileges for applications and users *should be* assigned out-of-band to an Ed-Fi REST API.

API Guidelines Contents

Find out more about the Ed-Fi API Design & Implementation Guidelines:

Application Authorization

Applications perform actions against data based on user directives. It is important to consider what actions are appropriate in relation to specific data. In some cases, it *may not* be appropriate for an application to act within a specific resource domain. For example, an application that only records attendance *should not* have authorization to access assessment information. In the same way, it *may not* be appropriate for an application to be authorized to act within all resource domains. For example, a SIS package used only at one school *should not* be authorized to work with resources at other schools.

Application credentials *should* be assigned out-of-band for each resource group and (if necessary) each organization appropriate to the application within an Ed-Fi REST API.

User Authorization

User authorization typically consists of a subset of actions available to an application. However, users may be authorized to perform different actions on different classes of objects, or on distinct objects (domains). An Ed-Fi REST API *should* provide user authorization in a manner consistent with the sensitivity of the data provided.

User Roles

A role is a set of operations able to be performed by a user. Roles are typically broad-grained, cross-cutting, and application-specific. A Teacher role identifies a set of operations available to an individual, but does not intrinsically limit those operations to a specific classroom.

The specific operations performed by a role such as a Teacher will be different from one application to the next. Thus, an Ed-Fi REST API implementation *should* support resource-oriented claims issued based on the meaning of a given role in the implementation's context. This allows security to be managed at a conceptual level (e.g., by assigning personnel to conceptual roles like Teacher or Principal) while also supporting a fine-grained assignment of permissions to those roles specific to an organization or application.

External User Authorization

Many existing external applications (like SISs) have business logic that limit the operations available to users based on the users' roles as well as the resource upon which the user is operating. In this scenario, the Ed-Fi REST API provides application authorization, and the client application provides user authorization.

In a two-legged OAuth scenario with a trusted partner application (such as a SIS), resource requests *should* be secured by scoping requests to specific education organizations such as states, regional service centers, local education agencies, or schools. Within the authorized scope, the applications would be able to access and modify all appropriate data.

Internal User Authorization

An Ed-Fi REST API implementation limits the operations available to users based on the user's roles and resources upon which the user is operating. Internal user authorization is preferred over external user authorization because it ensures a consistent security model across all applications and simplifies the application vetting process.

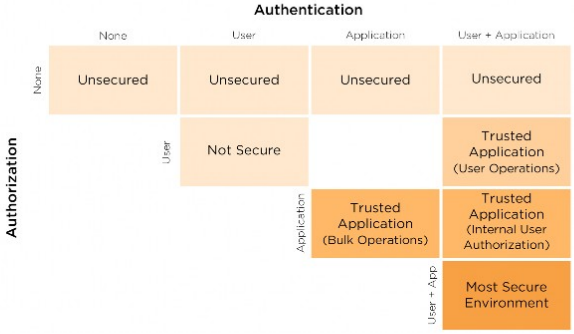
In order to communicate visual cues to users regarding their authorization, user authorization *must* be communicated to applications. Authentication claims *may* be enhanced to reflect high-level application-specific permissions using an out-of-band permission look-up. In an implementation of an Ed-Fi REST API, the HTTP OPTIONS method[12] *should* be implemented to provide resource-specific authorization information to applications. In any case, where operations are not allowed on a resource, the "403 Forbidden" response *should* be issued for unauthorized requests if the application and user authentication is otherwise valid.

Client Applications

For three-legged OAuth scenarios, each request *should* be authorized based on claims-based security (see below) and Ed-Fi domain data to identify the students for which they have responsibility. For example, superintendents would be granted access to student data for all students in their districts, principals would be granted access to all students in their schools, and teachers would be granted access only to students enrolled in their sections.

Authentication and Authorization Permutations

Authorization presupposes authentication. In the figure below, the empty boxes represent impossible security models. The lightest boxes (on the upper left) are less secure combinations and *should not* be used. The medium boxes are security models requiring that applications be certified for their intended purposes and *should* be used only with extreme caution. The darkest box (on the lower right) represents the *recommended* Ed-Fi REST API Authorization and Authentication model in which both the user and application are authenticated and authorized within the implementation.



	None	User	Application	User + Application
None	Unsecured	Unsecured	Unsecured	Unsecured
User		Not Secure		Trusted Application (User Operations)
Application			Trusted Application (Bulk Operations)	Trusted Application (Internal User Authorization)
User + App				Most Secure Environment

Figure 2. Permutations of Authentication and Authorization

[10] Specification details can be found in Hammer-Lahav, et al. (2011), The OAuth 2.0 Authorization Protocol, IETF Draft.

[11] For more information on STS, see here.

[12] For more information on the HTTP OPTIONS method, see Section 9.2 here.