AccessControl Web Service Design

Version 1.0

# Description

AccessControl is Auxiliary Web Service to trace user sessions and resolve authorization queries within Blueprint services. AccessControl is the internal part of enterprise solution infrastructure and service endpoint is not visible to client web application.

# Scope

AccessControl is limited to resolving session token into user information and permission set. AccessControl Web Service will be providing session management functionality for Blueprint application. The following design principles are used to limit the scope of the AccessControl Web Service for the current version:

1. Session Token is GUID represented via 32 alphanumerical characters passed via HTTP header Session-Token.
2. IIS service supports PUT and DELETE HTTP methods, alternatively X-HTTP-Method-Override header to be used in conjunction with POST method.
3. User session is considered expired after SESSION\_TIMEOUT (20 min) after last request sent by web browser to any of services.
4. Service is restartable.

# Interface

## Endpoint

Service will be hosted in the path /svc/accesscontrol

## Methods

GET /sessions

Method to query if session exists, expect to receive session token in header Session-Token to identify user session. Method will not extend lifetime of the session by SESSION\_TIMEOUT. This method to be used for sign in sequence only. Session information is returned back.

Returns status **200 OK** if user session exists and user is permitted to perform the operation on the artifact as specified.  
Returns status **401 Unauthorized** if session exists but user is not allowed to perform the operation on the artifact as specified.  
Returns status **403 Bad Request** if session, operation or artifact identifiers are not recognized.  
Returns status **404 Not Found** if session, operation or artifact not found.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Sessions not found.

Example: PUT /svc/accesscontrol/sessions/read\_artifact/123

PUT /sessions[/{op}[/{artifactId}]]  
- where {op} – optional parameter to identify operation user indents to perform, {artifactId} – optional parameter to identify artifact operation is requested to be performed on.

Method expect to receive session token in header Session-Token to identify user session. Method will extend lifetime of the session by SESSION\_TIMEOUT. This method to be used by all service methods to authorize user session. Session token value is returned back via Session-Token header.

Returns status **200 OK** if user session exists and user is permitted to perform the operation on the artifact as specified.  
Returns status **401 Unauthorized** if session exists but user is not allowed to perform the operation on the artifact as specified.  
Returns status **403 Bad Request** if session, operation or artifact identifiers are not recognized.  
Returns status **404 Not Found** if session, operation or artifact not found.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Sessions not found.

Example: PUT /svc/accesscontrol/sessions/read\_artifact/123

### POST /sessions/{userId}

- where {userId} – parameter to identify user for whom session needs to be created.

Method for initiating user session. Session Token is returned thru Session-Token header as the string containing 32 alphanumerical characters of unique identifier (GUID).

Returns status **200 OK** if session was initiated successfully.  
Returns status **401 Unauthorized** if request cannot be performed due to inadequate permissions.  
Returns status **403 Bad Request** if user identifier is missing or malformed.  
Returns status **404 Not Found** if user is not found for the identifier provided.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Sessions not found.

Example: POST /svc/accesscontrol/sessions/1234

DELETE /sessions

Method for removing user session upon explicit sign out. Method expect to receive session token in header Session-Token to identify user session. The session token is returned thru Session-Token header.

Returns status **200 OK** if no issue is detected.  
Returns status **401 Unauthorized** if request cannot be performed due to inadequate permissions.  
Returns status **403 Bad Request** if session token is missing or unrecognized.  
Returns status **404 Not Found** if session is not found for the token provided.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Sessions not found.

Example: DELETE /svc/accesscontrol/sessions

GET /sessions/select?ps={ps}&pn={pn}  
- where {ps} – page size , and {pn} – page number, 1-based.

Method returns paged list of existing sessions and max page number for the page size specified in the header. Methods expect to receive session token in header Session-Token to identify admin user session.

Returns status **200 OK** if admin user session exists and user is permitted to list sessions.  
Returns status **401 Unauthorized** if session exists but user is not allowed to list sessions, as only Application Admin group members are allowed to use this method.  
Returns status **403 Bad Request** if session, operation or artifact identifiers are not recognized.  
Returns status **404 Not Found** if session, operation or artifact not found.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Sessions not found.

Example: GET /svc/accesscontrol/sessions?ps=20&pn=1

GET /status  
Method to return current status of AccessControl Web Service.

Returns status **200 OK** if no issue is detected.  
Returns status **401 Unauthorized** if user is not permitted the action requested.  
Returns status **403 Bad Request** if session token is missing or unrecognized.  
Returns status **500 Internal Server Error** if error:  
- database connection failed;  
- database connection established but table Sessions not found.  
Returns status **503 Service Unavailable** if service is not ready.

# Technology

AccessControl Web Service will be implemented using ASP.NET Web API/C#. Database will be Microsoft SQL Server 2012.

# Behavior

## Methods

GET /sessions/{ps}/{pn}  
Method retrieves session information. Return value should not be cached. Method will not change the state of the sessions it returns. **This method should not be used in any regular non-administrative subroutine.**

PUT /sessions[/{op}[/{id}]]  
Method checks if session exists for the token provided in memory cache. If not found Sessions table is checked. If found then user authorization check is performed for operation on the artifact specified in optional parameters. Method extends user session by SESSION\_TIMEOUT mins.

### POST /sessions/{id}

Method deletes session if already exists for the user specified by parameter. Method creates new session record for the user specified by parameter in database table and memory cache. Session token is generated in the database using NEWSEQUENTIALID() and returned as a 32 alphanumerical character string in Session-Token header.

DELETE /sessions  
Method deletes session if exists.

GET /sessions/select?ps={ps}&pn={pn}  
Method retrieves paged list of current sessions. Return value should not be cached. Method will not change the state of the sessions it returns. **This method should not be used in any regular non-administrative subroutine.**

GET /status  
Method tries to execute SELECT COUNT(\*) FROM [SESSIONS]; to test database connection. Checks if service is ready for operation.

# Storage

Records to be stored in AdminStore.Sessions (database AdminStore, table Sessions)

## Columns

UserId : Int32, PK  
SessionId : uniqueidentifier, IX, default = NEWID()  
BeginTime : datetime, nullable, BeginSession sets current time (UTC), EndSession sets null  
EndTime : datetime, nullable, EndSession sets current time (UTC), BeginSession sets null