Runtime Infrastructure - Feature #12191

request client-principal and realm hook for REST, WEB and SOAP services

11/16/2026 03:42 PM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**reviewer:**	
vendor_id:	GCD	**production:**	No
case_num:		**env_name:**	
version_reported:		**topics:**	
version_resolved:			

Description

History

#1 - 11/16/2026 03:42 PM - Greg Shah

Problem

Applications written for PASOE commonly rely on PASOE's security layer to authenticate each web request and attach a CLIENT-PRINCIPAL to it. The 4GL code then reads that principal and makes it the session's identity:

```
hCP = SESSION:CURRENT-REQUEST-INFO:GetClientPrincipal().
IF VALID-HANDLE(hCP) AND hCP:VALIDATE-SEAL(cDomainKey) THEN
   SECURITY-POLICY:SET-CLIENT(hCP).
```

The user validation itself is often implemented in 4GL as an OERealm class (Progress.Security.Realm.IHybridRealm: ValidateUser, ValidatePassword, GetAttribute, GetAttributeNames, SetAttribute, RemoveAttribute), which PASOE calls during authentication.

FWD already implements OERequestInfo:GetClientPrincipal() and SetClientPrincipal(), and carries a principal with appserver requests, including one supplied by an Open Client. For REST, WEB and SOAP requests, FWD authenticates with its own web service security, but it does not currently build a principal from that authentication, and it does not call a 4GL realm class.

Work

1. After FWD authenticates a REST, WEB or SOAP request, build a CLIENT-PRINCIPAL for it (user, domain, roles, login time, session id), seal it with the configured domain access code, and return it from SESSION:CURRENT-REQUEST-INFO:GetClientPrincipal() for the duration of the request.
2. Add an authentication hook that can delegate user validation to a 4GL class implementing Progress.Security.Realm.IHybridRealm, configured in the directory, so an application's existing realm code can be reused. Call it with the same arguments and sequence as PASOE does, and map its result and attributes (including roles) into the principal.
3. Map roles from the principal to FWD ACLs, so access rules written as role requirements on URL patterns can be expressed in the FWD configuration.
4. Check the related built-ins against OpenEdge in this flow: REGISTER-DOMAIN with domain access codes, SEAL, VALIDATE-SEAL, SET-DB-CLIENT, SECURITY-POLICY:SET-CLIENT and GET-CLIENT.
5. Document the configuration in Securing the Web Services (REST SOAP WEBHANDLER).

Testcases

- REST, WEB and SOAP requests with valid and invalid credentials: the principal returned by GetClientPrincipal(), its attributes and seal, compared with PASOE for the same realm class.
- A realm class that rejects a user, accepts with roles, and sets attributes.
- A request without authentication on an anonymous service: GetClientPrincipal() returns what PASOE returns.
- Role-based ACL allowing and denying a URL pattern.