CHAPTER 4

Policy Decision and Enforcement Subcontract

The Policy Decision and Enforcement Subcontract defines the interactions between container policy enforcement points and the providers that implement the policy decisions required by Java EE containers.

4.1 Policy Enforcement by Servlet Containers

Servlet containers must employ the methods defined in the following subsections to enforce the authorization policies established for web resources.

4.1.1 Permission Names for Transport and Pre-Dispatch Decisions

The name of the permission checked in a transport or pre-dispatch decision must be the value that would result from applying the Servlet welcome file processing rules to the unqualified request URI minus the context path. For the special case where this transformation of the request URI yields the URLPattern "/", the empty string URLPattern, "", must be used as the permission name. The welcome file processing rules are defined in the Servlet specification.

For the special case where the empty string must be substituted for the "/" pattern in the permission evaluation, all target related processing (including servlet mapping, filter mapping, and form based login processing) must be performed using the original pattern, "/".

4.1.2 Evaluation of Transport Guarantees

The Servlet container must obtain a WebUserDataPermission object with name obtained from the request URI as defined in Section 4.1.1, "Permission Names for Transport and Pre-Dispatch Decisions". The actions of the obtained permission must be composed of the HTTP method of the request and a protection value describing the transport layer protection of the connection on which the request arrived. The protection value must be as follows:

- If the request arrived on a connection deemed by the container to be protected for confidentiality, a protection value of ":CONFIDENTIAL" must be used.
- If the request arrived on a connection deemed by the container to be protected for integrity (but not confidentiality), a protection value of ":INTEGRAL" must be used.
- If the request arrived on a connection deemed by the container to be unprotected, the actions used in the permission construction must contain only the HTTP method of the request.

The Servlet container must use one of the methods described in Section 4.7, "Checking AccessControlContext Independent Grants" to test if access to the resource using the method and connection type encapsulated in the WebUserDataPermission is permitted. If a SecurityException is thrown in the permission determination, it must be caught, and the result of the determination must be that access to the resource using the method and connection type is not permitted. If access is not permitted, the request must be redirected as defined by the Servlet Specification. If access is permitted, the request must be subjected to a pre-dispatch decision.

4.1.3 Pre-dispatch Decision

The Servlet container must obtain a WebResourcePermission object with name obtained from the request URI as defined in Section 4.1.1, "Permission Names for Transport and Pre-Dispatch Decisions". The actions of the obtained permission must be the HTTP method of the request. The Servlet container must use one of the methods described in Section 4.8, "Checking the Caller for a Permission" to test if the WebResourcePermission has been granted to the caller. If a SecurityException is thrown in the permission determination, it must be caught, and the result of the determination must be that the permission is not granted to the caller. The Servlet container may only dispatch the request to the web resource

if the WebResourcePermission is determined to be granted to the caller. Otherwise the request must be rejected with the appropriate HTTP error message as defined by the Servlet Specification.

Before it dispatches a call to a web resource, the container must associate with the call thread an AccessControlContext containing the principals of (only) the target component's runAs identity (as defined in Section 4.5, "Component runAs Identity).

4.1.4 Application Embedded Privilege Test

When a call is made from a web resource to `isUserInRole(String roleName)` the implementation of this method must obtain a WebRoleRefPermission object with name corresponding to the servlet-name of the calling web resource and with actions equal to the roleName used in the call. For the special case where the call to `isUserInRole` is made from a web resource that is not mapped to a Servlet (i.e. by a `servlet-mapping`), the name of the WebRoleRefPermission must be the empty string. In either case, the implementation of the isUserInRole method must then use one of the methods described in Section 4.8, "Checking the Caller for a Permission" to determine if the WebRoleRefPermission has been granted to the caller. If a SecurityException is thrown in the permission determination, it must be caught, and the result of the determination must be that the permission is not granted to the caller. If it is determined that the WebRoleRefPermission has been granted to the caller, isUserInRole must return true. Otherwise the return value must be false.

4.2 Provider Support for Servlet Policy Enforcement

In support of the policy enforcement done by servlet containers, providers must implement the policy decision functionality defined in the following subsections.

4.2.1 Servlet Policy Decision Semantics

A Policy provider must use the combined policy statements of the default policy context (as defined in Section 4.10, "Default Policy Context") and of the policy context identified by calling PolicyContext.getContextID to determine if they imply the permission being checked. If one or more excluded policy statements imply the checked permission, the evaluation may terminate and the checked permission must be determined not to be granted. Otherwise, if one or more unchecked policy statements imply the checked permission, the checked