



**Figure 3.1** PolicyConfiguration State Table

### 3.1.1.2 Linking Policy Contexts

In the Java EE security model, principal-to-role mappings have application scope; that is, the same principal-to-role mappings must apply in the access decisions applied at all of the modules (that may represent separate policy contexts) that comprise an application. Same application policy contexts must be associated by calling the `PolicyConfiguration.linkConfiguration` method. This method must create a transitive and symmetric relationship within the provider and between this `PolicyConfiguration` and the argument `PolicyConfiguration`, such that they and all `PolicyConfiguration` objects otherwise linked to either of them share the same principal-to-role mappings. The semantics of the association must preserve the invariant that at most one principal-to-role mapping may apply to any `PolicyConfiguration`.

### 3.1.2 Servlet Policy Context Identifiers

Servlet requests may be directed to a logical host using various physical or *virtual host* names or addresses, and an application server may be composed of multiple logical hosts. A virtual application server may be realized as a *cluster* of physical application servers, each hosting some subset of the logical hosts of the virtual application server. This specification uses the term *hostname* to refer to the name of a logical host that processes Servlet requests. A servlet container is responsible

for mapping the target name or address information of an HTTP request to the appropriate hostname.

To satisfy this specification, an application server must establish servlet policy context identifiers sufficient to differentiate all instances of a web application deployed on the logical host or on any other logical host that may share the same policy statement repository. One way to satisfy this requirement is to compose policy context identifiers by concatenating the hostname with the context path (as defined in the Servlet specification) identifying the web application at the host.

When an application is composed of multiple web modules, a separate policy context must be defined per module. This is necessary to ensure that url-pattern based and servlet name based policy statements configured for one module do not interfere with those configured for another.

In Servlet containers that support the programmatic registration and security configuration of servlets (e.g., Servlet 3.0 compatible Servlet containers), the policy contexts assigned to web applications and web modules must be distinct from those to which any EJB components are assigned.

### 3.1.3 Translating Servlet Deployment Descriptors

A reference to a `PolicyConfiguration` object must be obtained by calling the `getPolicyConfiguration` method on the `PolicyConfigurationFactory` implementation class of the provider configured into the container. The policy context identifier used in the call to the `getPolicyConfiguration` method must be a `String` composed as described in Section 3.1.2, “Servlet Policy Context Identifiers,” on page 23. The `security-constraint` and `security-role-ref` elements in the deployment descriptor must be translated into permissions and added to the `PolicyConfiguration` object as defined in the following sections. Before the translation is performed, all policy statements must have been removed<sup>2</sup> from the policy context associated with the returned `PolicyConfiguration`.

#### 3.1.3.1 Programmatic Servlet Registrations

In Servlet containers that support the programmatic registration and security configuration of servlets (e.g., Servlet 3.0 compatible Servlet containers), the

<sup>2</sup> This can be achieved by passing `true` as the second parameter in the call to `getPolicyConfiguration`, or by calling `delete` on the `PolicyConfiguration` before calling `getPolicyConfiguration` to transition it to the open state.