HRAC RFP Submission: Outstanding Issues

Title	**Quality of Protection as an authorization decision factor**
ID	6
Priority	2
Description	Should current quality of protection policy information in ADO client be used as a factor in authorization decisions as principal credentials are?
Date Issued	8/10/98
Depends on Issues	No dependencies
Pointed by	Konstantin Beznosov
Related Refs	msg00055.html -- msg00057.html
To propose a resolution	
Proposed Resolution	

Title	**All policy evaluators have to be consulted**
ID	36
Priority	2
Description	Since the policy combinator is handed only results of the policy evaluation, all PolicyEvaluators have to be consulted before a final decision can be made. The work of DecisionCombinator cannot be optimized. For example, even in the case when the very first decision from a PolicyEvaluator makes the "final call" all other PolicyEvaluators have to be consulted.
Date Issued	11/10/98
Depends on Issues	No dependencies
Pointed by	Polar Humenn
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	Agree that this is an issue - we'll think about it.

Title	**Grouping of resources for associating them with policies**
ID	37
Priority	2
Description	How can resources of a particular types to be associated with particular policy or policies?
Date Issued	11/10/98
Depends on Issues	No dependencies
Pointed by	Ed from NIST
Related Refs	No additional references
To propose a resolution	Konstantin
Proposed Resolution	In order to solve this problem we need a "type" associated with ResourceName(s) we will look at providing a * wildcard in the value field of the admin interface whose semantics is any value of this attribute Do we also need to do it in PolicyEvaluatorAdmin as well as Policy EvaluatorLocatorAdmin

Title	**Removing safely PolicyEvaluator**
ID	39
Priority	2
Description	How can a client of an implementation of the PolicyEvaluatorLocatorAdmin interface safely remove an evaluator? The interface does not have a delete_evaluator(s) operation, so I would assume that the recommended technique would be to: 1. Get sequence of evaluators via PolicyEvaluatorLocator::get_policy_decision_evaluators. 2. Build a new PolicyEvaluatorList removing the unwanted evaluator. 3. Invoke PolicyEvaluatorLocatorAdmin::replace_evaluators with the new list. The problem is what happens when another client invokes add_evaluators or replace_evaluators in the time frame between steps 1 and 3 above. I would suggest that other client will be quite unhappy.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Bob Burt
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**safely restoring a list of default evaluators**
ID	40
Priority	2
Description	How can a client of an implementation of the PolicyEvaluatorLocatorAdmin interface safely restore a list of default evaluators? Since the PolicyEvaluatorLocatorAdmin::set_default_evaluators does not return a list of the current default evaluators, it will be impossible to reliably return the list of default evaluators to their previous state. (Trying to get the default list by invoking get_policy_decision_evaluators with a bogus resource name seems to have the same "non-locking problem" as described in Issue 39
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Bob Burt
Related Refs	issue #39
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**safely removing a DecisionCombinator for a given resource name**
ID	41
Priority	2
Description	How can a client of an implementation of the PolicyEvaluatorLocatorAdmin interface safely remove a DecisionCombinator for a given resource name? There appears to be no way to do this.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Bob Burt
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**safely restoring the default DecisionCombinator**
ID	42
Priority	2
Description	How can a client of an implementation of the PolicyEvaluatorLocatorAdmin interface safely restore the default DecisionCombinator? This issue is similar to Issue 40.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Bob Burt
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**invalid object reference for a PolicyEvluator and/or DecisionCombinator**
ID	43
Priority	2
Description	What is an implementation of the AccessDecision interface supposed to do when the PolicyEvaluatorLocator returns an invalid/unreachable object reference for a PolicyEvluator and/or DecisionCombinator? The specification makes no demands on implementations of the PolicyEvaluator and DecisonCombinator interfaces. That is, it does not dictate whether they must be "transient" or "persistent", whether they must maintain state or not, and whether they can be launched by an ORB or must be manually launched. For example, based on the specification, it is perfectly legal to build an implementation of the PolicyEvaluator interface that is “transient” and must be manually launched. This means the IOR for this implantation is only valid for the life of the process that created it and the process can only be manually launched. If the process is killed and its clients (e.g. ADO’s) continue to try to use it, they will get CORBA System Exceptions (COMM_FAILURE in OrbixWeb). The client has no way of determining that the object reference is valid or invalid, the process is running or not running, or whether the network path is available or not available. The real issue here seems to be that the client (e.g. ADO) has no way of notifying the PolicyEvaluatorLocator that a problem exists and that perhaps it should investigate the fact that it is distributing a potentially bogus object reference.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Bob Burt
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**state persistently in PolicyEvaluatorLocatorAdmin and PolicyEvaluatorAdmin**
ID	44
Priority	2
Description	Clients of the PolicyEvaluatorLocatorAdmin and PolicyEvaluatorAdmin interfaces cannot determine whether or not the implementations of those interface maintain state persistently. This is important to a client because it must know whether of not it needs to reset the state to that which it expects. Without this knowledge, the client code would need to be changed to migrate from a persistent to a non-persistent implementation. If the client code must change between implementations, why have a specification?
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Bob Burt
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**scalability of the proposed spec implementations**
ID	45
Priority	2
Description	Can complete implementations of this specification actually be made to be "scalable"? From the experience that I have had, users of security control systems expect minimal if not transparent overhead from such a system. This is often not considered a problem during the design phases, but when implementations begin to adversely affect response times or overall use of system resources ($$$), the problem becomes real. When one looks at the Access Decision Model (2.3.1), it appears that the Access Decision will make a minimum of four operation invocations. This in itself could result in unacceptably excessive overhead. Even more importantly, one should consider what would comprise an implementation of a PolicyEvaluatorLocator interface. This interface must maintain a collection of “resource name – evaluator” associations. Each time that it returns evaluators, it must query this collection to determine if the requested resource name matches one or more of those “resource name – evaluator” associations. It must query this collection for every request for a list of evaluators. If this collection is small enough to be held in its entirety in memory, this can be a somewhat compute-intensive process. If, however, the system has scaled to the extent that this collection can no longer be effectively maintained in memory, one must now endure to extremely high overhead of doing external storage based queries. Perhaps one could invent some form of caching technique that might optimize this query; however, it appears to me that this might not be possible. It appears that the "entire" collection must be queried to ensure that there are no "resource name -- evaluator" associations. Implementation of these external storage based queries would seem to create overhead that would be unacceptable to all but the least demanding users.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Bob Burt
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**Dropping the requirement for defining a policy object interface**
ID	46
Priority	2
Description	Issue #1 Choosing not to address this requirement results in almost all of the rest of my objections. It seems to me that the goal of the RFP is to create a standard interface for the evaluation of a security policy. Necessitated by that is the standardization of a policy object interface. Please note that I am NOT proposing the standardization of policy CONTENT! Clearly, that would be impossible. However, the interface should be definable, and the capabilities that can be defined/controlled within a policy should be definable. As a result of not defining this policy interface, you've instead had to define a lot of pieces of what ought to be implementation details, and proposed that the industry should standardize on the mechanism for evaluating policy.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Kurt Schurenberg
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**The lack of a Context Sensitive ACL makes the submission ineffective**
ID	47
Priority	2
Description	Issue 2. The lack of a Context Sensitive ACL makes the submission ineffective. (It's a weakness of the RFP that Context-Sensitive ACL is optional, IMHO.) Context Sensitive ACL is an optional requirement which appears to be unaddressed as a result of the decision to not support the required requirement for policy interface definition. If the submission dealt with objects as objects, then the policy interface could allow decisions based on the contents of the object being evaluated. Since the submission has moved instead to controlling access to named strings, there is no option except to evaluate each attribute of an object which might be sensitive or policy-driving as a separate named resource. This creates more load on a system which appears to be likely to have performance implications already. Imagine a system which allows retrieval of patient records from a variety of "publishers" (like hospitals, clinics, practices, etc.) A policy may allow a user to see any patient record published by a hospital or clinic with which the user is associated, unless the diagnosis of the record has to do with AIDS, mental health, or alcoholism. By it's very nature, the evaluation is done after data has been retrieved (but before it is shown to the user), so the filtering cannot be applied on the front end (by modifying the query, for example). Thus, you must look inside the contents of the object on which policy is being evaluated to find data relevant to the decision being made. The Dynamic attributes do not provide a method for defining this, and since the object itself is not available, there's no way to make decisions based on this. If it's thought that you can define the interface to hand in this "sensitive" data, configured somehow as a security attribute, this means that the policy must be embedded in the code which queries the system for authorization. That would be an expensive and difficult system to maintain.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Kurt Schurenberg
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**Policy Evaluators are not interchangeable**
ID	48
Priority	2
Description	Issue 3. Since Policy is not defined, the idea that Policy Evaluators are interchangeable seems disingenuous. It seems that if policy interface is non-standard, then policy evaluators are non-standard, by definition. Thus, the proposed standard for interoperation doesn't succeed in providing interoperability.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Kurt Schurenberg
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**no extra capabilities than what CORBA security has**
ID	49
Priority	2
Description	Issue 4. Little or no new capabilities appear to be provided. I understand that this specification is designed to work with or without an existing CORBA security layer. However, it would appear to me that what the specification does is rework the CORBA security capabilities to produce the same capabilities in a different way. That different way includes a loss of object orientation due to use of named resources, which seems like a negative to me. There doesn't appear to be an ability to do any more than was already available via the standard CORBA security. Since you aren't using OO for policies or the objects controlled, but just name strings, it's possible that things can be defined at a different level here, but I'm not sure that's a positive change.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Kurt Schurenberg
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**Decision Combinator overtakes things which ought to be defined in policy**
ID	50
Priority	2
Description	Issue 5. The Decision Combinator overtakes things which ought to be defined in policy. It seems to me that a security policy defines how to combine multiple incoming elements of an authorization decision. When you add the capabilities to modify the decision combinator to make more complex grouping and evaluation of individual policy evaluation results, you will have started to define the policy interface that you are trying to avoid defining. Again, though, you are starting to require that the policy be embedded in code, rather than metadata, which makes the system extremely expensive to maintain.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Kurt Schurenberg
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**Specified system requires high overhead**
ID	51
Priority	2
Description	Issue 6. System appears to require a high overhead. By ignoring the admittedly difficult issue of defining a policy interface, the submission has been forced to create a complex set of objects which may be evaluating policy on details down to the attribute level. If a search has produced a list of objects, each of which contain 3 attributes which are policy-controlled, then you must go through the 3 object interfaces four times for every element on the list. This will be too slow to be commercially acceptable.
Date Issued	1/21/99
Depends on Issues	No dependencies
Pointed by	Kurt Schurenberg
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**ADO interfaces Exceptions**
ID	2
Priority	3
Description	What exceptions should be raised by ADO's methods? Should it be the matter of a policy whether ADO raises an exception when something goes wrong or silently denies access to a resource? Three possible directions are identified: 1. Methods raise no exceptions 2. Methods raise exceptions a. Methods raise only system exceptions (like NO_PERMISSION, BAD_PARAM, NOT_IMPLEMENT) b. Methods raise system and application exceptions,
Date Issued	8/11/98
Depends on Issues	
Pointed by	Konstantin Beznosov
Related Refs	mail list archive messages # msg00040.html, msg00054.html
To propose a resolution	
Proposed Resolution	

Title	**Exception(s) raised by multiple_action_access_allowed() method in ADO interface**
ID	4
Priority	3
Description	From her message: "Should access decision methods throw exceptions at all... an audit log should have this info... but not the client... seems it should be a binary decision." Derived from a conference call discussion: How would a programmer use an exception returned by multiple_action_access_allowed() method? Is not it better return any problem indications in the returned sequence instead of raising an exception?
Date Issued	8/10/98
Depends on Issues	2
Pointed by	Carol Burt
Related Refs	
To propose a resolution	
Proposed Resolution	

Title	**Consistent Terminology**
ID	7
Priority	3
Description	Can we define some consistent Terminology? Define Evaluator and Policy Name
Date Issued	8/10/98
Depends on Issues	11, 12, 13
Pointed by	Carol Burt / John B.
Related Refs	msg00039.html
To propose a resolution	David Chizmadia to fill in secti
Proposed Resolution	

Title	**Correct name of the specified functionality: no "access control" but "authorization decisions"**
ID	18
Priority	3
Description	"HRAC" stands for healthcare resource access control. Clearly, the functionality for which the RFP is asking (and what a submission is supposed to specify) is concern only with making authorization decisions, i.e. no actual access control is in the scope of the RFP. Thus, the specified functionality should be renamed from "access control" to something else that would reflect the fact that it specifies only authorization decision part, i.e. not control.
Date Issued	8/11/98
Depends on Issues	No dependencies
Pointed by	Konstantin Beznosov
Related Refs	msg00110.html
To propose a resolution	
Proposed Resolution	

Title	**Facility or Service**
ID	21
Priority	3
Description	Should the final functionality be called a "facility", "service", or something else?
Date Issued	8/11/98
Depends on Issues	No dependencies
Pointed by	Konstantin Beznosov
Related Refs	No additional references
To propose a resolution	Nobody Assigned
Proposed Resolution	

Title	**Use cases for the proposed interfaces**
ID	25
Priority	3
Description	The submission needs use cases to: 1. To see if the proposed design will satisfy most of the cases of the facility use, 2. To show how the proposed facility should be used. It would be nice to do a use case from COAS
Date Issued	9/11/98
Depends on Issues	No dependencies
Pointed by	
Related Refs	No additional references
To propose a resolution	John Barkley and Konstantin
Proposed Resolution	John will review and modify introductory text

Title	**ASTM Access Control Matrix security standard**
ID	26
Priority	3
Description	"Has anyone looked at the ASTM standard? It would be most PC to include a reference...if it applies."
Date Issued	10/ 6/98
Depends on Issues	No dependencies
Pointed by	Mary Kratz
Related Refs	E-mail message from Mary Kratz to the submission list on 10/1/98
To propose a resolution	Konstantin
Proposed Resolution	Konstantin will contact Mary Kratz to get a copy for review