GENI

Global Environment for Network Innovations

Distributed Identity and Access Control (DIAC) API

Spiral 1 Draft 0.9

Document ID: GENI-DIAC-API-0.9

March 12, 2010

Prepared by Jay Jacobs and Stephen Schwab

SPARTA, Inc. d.b.a. Cobham Analytic Solutions

Table of Contents

1. Scope

1.1 Purpose

This document describes the Distributed Identity and Access Control (DIAC) web service API and builds on the concepts described in the GENI-RBAC-REQ document. Over the course of development and prototyping spirals this document will include further extensions for Attribute-Based Access Control (ABAC) in the context of slice-based control frameworks and their security policies. DIAC uses ABAC technology, which is a form role-based access control (RBAC), for access control and automated trust negotiation. The term ABAC refers specifically to the SPARTA Attribute-Based Access Control implementation, and the approach, as introduced by Will Winsborough et al. in [WJ03a]. As a reader or developer, your comments, criticisms, and suggestions are welcome and essential to progress.

1.2 Background

As shown in Figure 1 - ABAC Usage, the ABAC web service is designed for access control of generic resources, managed by a service provider and used by the requestor. Trust negotiators are peers as shown in Figure 2 - Trust Negotiation. Trust negotiations are cached for optimization, so the access mediator only needs to negotiate with the requestor's peer initially and when credentials expire. Peers may be location independent but should be authenticated.

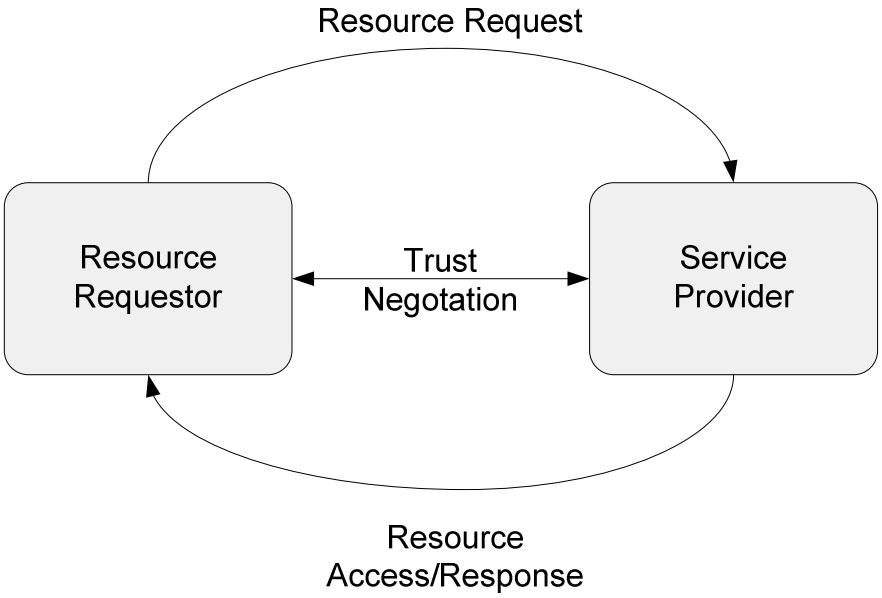

Figure 1 - ABAC Usage

The ABAC web service employs a request-response message format for each operation. Basic operations allow for the creation and management of negotiation contexts, access requests, and internal trust negotiation. Section 2 describes the external messages needed for access requests and managing contexts. Section 3 describes the trust negotiation functions used internally by ABAC. Most message type fields are strings or URLs. Appendix A describes the remaining complex data types, and Appendix B describes the usage of the API. Service providers in ProtoGENI mediate access control at decision control points such as component managers and slice authorities. Requestors represent authenticated ProtoGENI principals.

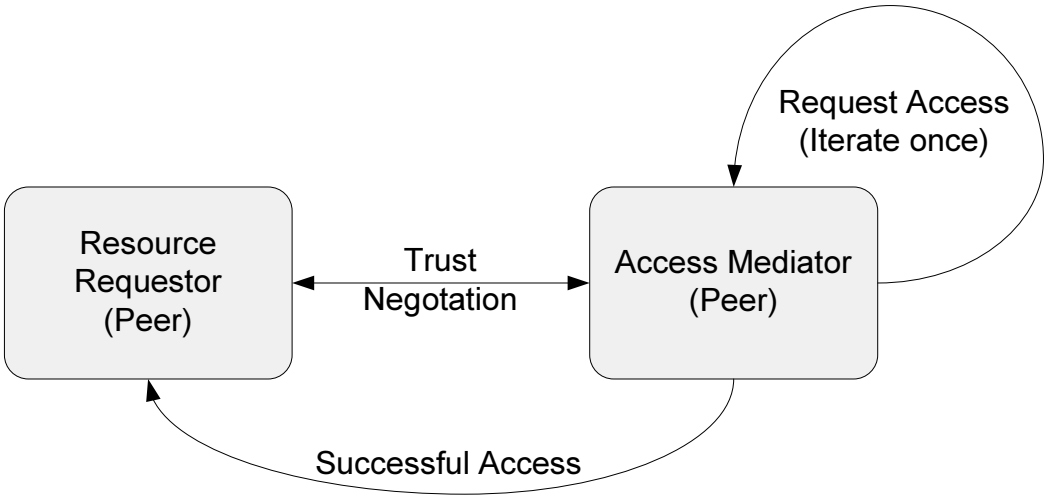

Figure 2 - Trust Negotiation

1.3 Related Documents

Some of the material in this document is drawn from the following documents listed below.

Document ID	Description
WJ03a	Automated Trust Negotiation Technology with Attribute-based Access Control, W. Winsborough and J. Jacobs, In Proceedings of the DARPA Information Survivability Conference and Exposition, 2003, Vol. 2 pp 60-62, April 22-24, 2003.
GENI-RBAC-REQ	RBAC Requirements for ProtoGENI, S. Schwab, J. Jacobs, and A. Hussain. January 15, 2010.
B64	The Base16, Base32, and Base64 Data Encodings, S. Josefsson, October 2006, http://tools.ietf.org/html/rfc4648

2. Basic Services

2.1 Access

Requests access a resource on behalf of a principal. This function should be made to the service provider's negotiator. Each access request is bound to a specific (pair of) negotiation context(s). Access requests should be initiated by a decision control point. If the local service possesses sufficient policy (credential sets), then a decision can be made without the need for a negotiation. If a negotiation is required, the access request specifies the web service location (local or remote) and negotiation context for both peers.

2.1.1 Request

peerURL - the URL for the peer service
selfURL - the URL for the current service
goal - the primary trust target for the negotiation

2.1.2 Response

goal - the primary trust target for the negotiation
result - success or failure
provenance - the trust target graph from the service provider's negotiator

2.2 CreateContext

Creates a new negotiation context in a negotiation service. Credentials cannot be removed from a context. This operation can be used to restore a context to a default setting.

2.2.1 Request

contextInfo - reference to a negotiation context as an XML object, file, or reference handle
peerURL - the URL for the peer service

2.2.2 Response

contextInfo - reference handle to a negotiation context
peerURL - the URL for the peer service

2.3 CredentialUpdate

Adds a set of ABAC credentials into a specific negotiation context.

2.3.1 Request

issuerCredentials - a set of credentials which can be traced to the issuer
subjectCredentials - a set of credentials which can be traced to the subject
traces - attributes names to ensure a negotiation can be completed
context - the negotiation context to update with these credentials

2.3.2 Response

results - success or failure for each credential

2.4 AddCertificate

Each web service maintains an internal cache of public key certificates for all issuers. The cache is used to verify credentials during the negotiation protocol. This function adds a new X.509 identity certificate to a negotiator service.

2.4.1 Request

certificate - PEM (base-64) encoded X.509 certificate

2.4.2 Response

alias - the SHA-1 hash of the public key contained in the X.509 certificate

2.5 RemoveCertificate

Removes an X.509 identity certificate from the shared identity cache. All credentials issued by the principal with the corresponding private key will be invalidated for all subsequent negotiations.

2.5.1 Request

alias - the identifier for the public key

2.5.2 Response

alias - the identifier for the public key

3. Negotiation Services

3.1 Discovery

The discovery service allows for searching the credentials by issuer, defining role, and subject, where the defining role is an attributed roles and the subject may be an attributed role or a principal entity. The discovery service originally provided is designed to be a reference implementation. Future versions may be integrated into other publicly available services.

3.1.1 Request

op - search by "issuer", "role", or "subject"

one of the following:

role - the defined attributed role for backward searching
subject - the subject for forward searching
issuer - the issuer for auditing

3.1.2 Response

op - search by "issuer", "role", or "subject"

one of the following:

role - the defined attributed role for backward searching
subject - the subject for forward searching
issuer - the issuer for auditing
result - a set of matching X.509 attribute certificates

3.2Negotiate

Negotiate is used to send trust target graph changes between negotiators for a specific negotiation context pair. The call is initiated by the services provider's agent and sent to the resource requestor's agent. The service provider incorporates the requestor's changes before its next iteration.

3.2.1 Request

messageType - automated trust negotiation message
contextSource - the source negotiation context
contextDest - the destination negotiation context
selfURL - the URL of the originating negotiator
oppoURL - the URL of the opponent (peer) negotiator

3.2.2 Response

messageType - automated trust negotiation message
contextSource - the source negotiation context
contextDest - the destination negotiation context
selfURL - the URL of the originating negotiator
oppoURL - the URL of the opponent (peer) negotiator

Appendix A: Complex Data Types

A.1 Trust Target

Trust targets are used in a trust target graph, which each negotiator maintains independently. The "goal" of a negotiation is the root node of a trust target graph. Identical graphs are constructed by negotiating peers for each access request. Trust targets have the following fields:

- *Verifier* – the entity who requires the target role be satisfied
- *Target Role* –the attributed role which is required for success
- *Subject*—the principal entity that needs to prove inclusion in the target role

A.2 Context Info

The ContextInfo data type represents a negotiation context as a reference, file path to a persistent context, or an XML representation of a context. Persistent context storage is a future optimization/caching feature. ContextInfo elements may have one of the following fields:

- *Reference*—a unique string identifier which should correspond to a peer/session
- *FilePath*—a relative or absolute path to a persistent negotiation context

Contexts store signed credentials and define negotiation strategies (e.g. depth-first, breadth-first, etc.)

A.3 Messages

Messages define the protocol for automated trust negotiations. Negotiators use messages to transmit iterative changes to their trust target graphs. Messages contain node operations, edge operations, and credentials, which are used as evidence for edge operations. Messages have the following array elements, all of which are optional:

- *NodeOp*—the node operations occurring during an iteration of graph processing
- *EdgeOp*—the edge operations occurring during an iteration of graph processing
- *Evidence*—the credentials needed for proving entailment of the edge operations contained within this message

A.4 Credentials

Credentials are signed X.509v2 attribute certificates, which are signed by the issuer but do not contain any cryptographic key material. Credentials are added to negotiation contexts and transmitted between peers using messages.

Appendix B: Usage

This section briefly describes the usage of the ABAC web services API, which is subject to optimization as the prototyping enables a better understanding of the usage and optimization of slice life-cycles. Negotiation contexts are the main mechanism for caching and may be reset on a per request basis or periodically. The following is a typical cycle of usage:

1. Create the identity certificates and private keys needed for an ABAC policy.
2. Generate ABAC credentials using the private key(s) of the issuer(s).
3. Add the identity certificates needed for a negotiation.
4. Add the issuer based credentials to the service-side negotiator (or discovery service)
5. Add the subject credentials to the requestor-side negotiator.
6. Decision control points issue an access request based on local policy.
7. Peer negotiators exchange messages until a decision is reached.
8. The decision is returned to the access mediator

Before any negotiations begin, identity certificates and private keys must be generated for all issuers and subjects. Credentials must also be signed by their issuers. ABAC relies on chains of credentials, so each identity certificate may be self-signed since each issuer acts as a traditional trust anchor. Note that local access mediator policies then specify the target roles required for principals. Credentials reside in contexts for the service provider (issuer traceable) and access requestor (subject traceable) negotiators. *Only credentials may be cached in ABAC and only while the credentials remain valid.* This means a trust target graph is constructed for each access request, which is initiated by the service provider (slice authority, component manager), which negotiates with a peer for the principal subject. Negotiation messages are transmitted between peers if a credential chain cannot be constructed locally. Finally, the service provider's peer returns the success or failure of the negotiation. The trust target graph embodies all successes and failures as the graph is processed.