Google Search Appliance

Policy ACL API Developer’s Guide

Google Search Appliance software version 7.2 and later

Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
www.google.com

GSA-ACLAPI_200.01
March 2015

Contents

Policy ACL API Developer’s Guide 4

Policy ACL API Developer’s Guide: Java 4

- Getting Started 4
- API Authentication 6
- Pattern ACL API 6
- Group and Member API 9

Policy ACL API Developer’s Guide: .NET 12

- Getting Started 12
- API Authentication 14
- Pattern ACL API 14
- Group and Member API 14

Policy ACL API Developer’s Guide: Protocol 16

- API Authentication 16
- Pattern ACL API 17
- Group and Member API 19

Policy ACL API Developer’s Guide

The Google Search Appliance Policy ACL API enables you to programmatically configure policy access control lists on a search appliance. You can use this API to add users or groups to a URL pattern to which you restrict access. The policy ACL software improves search appliance performance by substantially reducing HEAD requests for user authorization information from remote servers.

This document provides the following topic areas:

- “Policy ACL API Developer’s Guide: Java” on page 4
- “Policy ACL API Developer’s Guide: .NET” on page 12
- “Policy ACL API Developer’s Guide: Protocol” on page 16

Policy ACL API Developer’s Guide: Java

The sections that follow specify policy ACL rules using Java.

Getting Started

The google-enterprise-gdata-api (http://code.google.com/p/google-enterprise-gdata-api) open source site provides ZIP files that contain the Java client library, source code and some sample applications for your reference. The information in this section helps you understand how to write your own applications based on the client library and how to run the provided open source sample applications. You can also use the sample applications as models for your own development.

Before starting, you need the following software:

- JDK version 5.0 or later available from Java.com (http://www.java.com/en/download/manual.jsp) downloads.
- Apache Ant (http://ant.apache.org/) version 1.7 or later.
- Admin Console user name and password for the search appliance to which you direct your commands.

After you download the software and acquire search appliance credentials, get started as follows:

1. Browse to the Administrative API download site (http://code.google.com/p/google-enterprise-gdata-api).

2. Download the ZIP file gsa-admin-api-java-1.0.0.zip (http://google-enterprise-gdata-api.googlecode.com/files/gsa-admin-api-java-1.0.0.zip) containing the client library and the sample application files.

3. Unzip the file and navigate to the `gdata/java` folder. The client library JAR files are present in the `lib` folder and the sample applications are present in the `sample` folder.

Running Samples

Sample applications are located in the `gdata/java/sample` folder.

To see if you've installed the required software correctly, open a command prompt and change directory to the `gdata/java` folder. Enter the following command:

```
ant -f build-samples.xml sample.dashboard.run
```

This command displays build output messages and opens an example dashboard that demonstrates the features of the Google Search Appliance Administrative API. You can add your search appliance configuration information to the dashboard and see the dashboard running.

To find out how to run each sample, go to `gdata/java/build-samples` folder and view the build file for each sample, like `dashboard.xml`, `commandline.xml`, and `aclclient.xml`. Look for the `samples run` comment. Another example of how to run the `commandline` sample application, enter the following command in the `gdata/java` folder:

```
ant -f build-samples.xml -Dargs="retrieve --hostname=gsa_hostname
--username=gsa_user --password=gsa_passwd --protocol=http
--port=gsa_port config crawlURLs" sample.commandline.run
```

Substitute these parameters:

- *gsa_hostname*—The host name of a search appliance (specify just the host name, not the domain name)

- *gsa_user*—The user name for the Admin Console on the search appliance

- *gsa_password*—The password for the Admin Console on the search appliance

This command displays the Crawl URLs settings on the search appliance.

The `commandline` sample application requires that command line arguments be passed to Ant by using the `-Dargs` option in the `commandline.xml` file.

Building Your Applications

You can build your own applications using the client library.

Copy the following client library JAR files from the `gdata/java/lib` folder to your development folder and add the files to your `classpath` environmental system variable:

- `gdata-core-1.0.jar`
- `gdata-gsa-1.0.jar`
- `gdata-client-1.0.jar`
- `gdata-client-meta-1.0.jar`
- `gdata-gsa-meta-1.0.jar`

You can then use the JAR files in your application.

API Authentication

Before making API calls with the Java client library, you must construct a new `GsaService` object or a `GsaClient` object. The `GsaClient` object provides a simplified interface to the functionality in `GsaService` object. Please note that for the Group and Member API (see "Group and Member API" on page 9), the `GsaClient` object cannot be used.

In the constructor that follows, replace *myUserId* and *myPassword* with your Admin Console authentication information:

```
// Creates GsaService object
GsaService service = new GsaService("google-adminfeed-1", "http", gsaAddr,
gsaPort);
service.setUserCredentials("myUserId", "myPassword");
// Or creates GsaClient object
GsaClient myClient = new GsaClient(gsaAddr, gsaPort, "myUserId", "myPassword"));
```

Pattern ACL API

The code in the sections that follow specifies the URL pattern for a rule.

Creating an ACL Rule

To create an ACL rule:

```
GsaEntry entry = new GsaEntry();
entry.addGsaContent("urlPattern", "http://example.com");
entry.addGsaContent("acl", "group:testGroup user:john");
myClient.insertEntry("policyAcls", entry);
```

Alternate method using protocol buffer encoding:

```
GsaEntry entry = new GsaEntry();
entry.addGsaContent("urlPattern", "http://example.com");
entry.addGsaContent(“protoAcls”, “true”);
entry.addGsaContent("acl",
"entries <\n” +
“  gsa_entry <\n” +
“    access: 1\n” +   // 1 is PERMIT access, 2 is DENY access
“    principal <\n” +
“      scope: 2\n” +  // 1 is USER scope, 2 is GROUP scope
“      name: \"testGroup\"\n” +
“      name_space: \"Default\"\n” +
“      case_sensitive: 0\n” +  // 0: all strings are case sensitive, 1: all
strings are case-insensitive
“    >\n” +
“  >\n” +
“>\n” +
“entries <\n” +
“  gsa_entry <\n” +
“    access: 2\n” +
“    principal <\n” +
“      scope: 1\n” +
“      name: \"john\"\n” +
“      name_space: \"Default\"\n” +
“      case_sensitive: 0\n” +
“    >\n” +
“  >\n” +
“>\n");
myClient.insertEntry("policyAcls", entry);
```

Protocol buffers are open source. For more information, visit http://code.google.com/p/protobuf/.

Here are the relevant protocol buffer message definitions:

```
// Information about the domain associated with the principal.
message Domain {
 enum DomainType {
   // Domain type used by most windows / active directory deployments.  Currently
this
   // is the only supported domain type.
   NETBIOS = 0;
 }
 required string name = 1;
 required DomainType type = 2 [default = NETBIOS];
}

// Information that fully specifies the user/group in the ACL.
message AclPrincipal {
 enum SCOPE {
   USER = 1;
   GROUP = 2;
 }

 enum CaseSensitivity {
   // All strings in AclPrincipal and its sub-messages are treated as
   // case-sensitive.
   EVERYTHING_CASE_SENSITIVE = 0;

   // All strings in AclPrincipal and its sub-messages are treated as
   // case-insensitive.
   EVERYTHING_CASE_INSENSITIVE = 1;
 }

 required SCOPE scope = 1;
 required string name = 2;
 optional string name_space = 3;
 optional Domain domain = 4;
 required CaseSensitivity case_sensitive = 5 [default =
EVERYTHING_CASE_SENSITIVE];
}

message GsaEntry {
 enum ACCESS {
   PERMIT = 1;
   DENY   = 2;
 }

 required ACCESS access = 2;
 required AclPrincipal principal = 5;
}

message GsaAclEntry {
 optional GsaEntry gsa_entry = 1;
}

message GsaAcl {
 repeated GsaAclEntry entries = 1;
}
```

Retrieving ACL Rules

To retrieve all ACL rules:

```
GsaFeed feed = myClient.getFeed("policyAcls");
for(GsaEntry entry : feed.getEntries()) {
  System.out.println("Url Pattern: " + entry.getGsaContent("urlPattern"));
  System.out.println("ACL rules: " + entry.getGsaContent("acl"));
}
```

To retrieve an ACL rule for a URL pattern:

```
GsaEntry entry = myClient.getEntry("policyAcls", "http://example.com");
System.out.println("Url Pattern: " + entry.getGsaContent("urlPattern"));
System.out.println("ACL rules: " + entry.getGsaContent("acl"));
```

To retrieve an ACL rule for a URL pattern in the protocol buffer format:

```
GsaEntry entry = new GsaEntry();
entry.addGsaContent("urlPattern", "http://abc2.com");
entry.addGsaContent("acl", "group:testGroup user:john");
myClient.updateEntry("policyAcls", "http://example.com", entry);
```

Updating an ACL Rule

To update an ACL rule:

```
GsaEntry entry = new GsaEntry();
entry.addGsaContent("urlPattern", "http://abc2.com");
entry.addGsaContent("acl", "group:testGroup user:john");
myClient.updateEntry("policyAcls", "http://example.com", entry);
```

Deleting an ACL Rule

To delete an ACL rule:

```
myClient.deleteEntry("policyAcls", "http://example.com");
```

Group and Member API

The code in the sections that follow specifies which users and groups are allowed to access a URL pattern that you set in the pattern ACL API.

Creating a Group

To create a new group:

```
// Create group "testGroup"
GsaEntry groupEntry = new GsaEntry();
groupEntry.addProperty("groupId", "testGroup");
service.insert(new URL("http://Search_Appliance:8000/a/feeds/group/2.0/domain/
"), groupEntry);
```

Retrieving Groups

To retrieve all groups:

```
GsaFeed groupFeed = service.getFeed(new URL("http://Search_Appliance:8000/a/
feeds/group/2.0/domain/"),
  GsaFeed.class);
for(GsaEntry groupEntry : groupFeed.getEntries()) {
  System.out.println("Group Name: " + groupEntry.getProperty("groupName"));
}
```

If the number of groups in the search appliance is more than 500, the result is represented on multiple pages—you can access the next page as follows:

```
if (groupFeed.getLink(Link.Rel.NEXT, Link.Type.ATOM) != null) {
  groupFeed = service.getFeed(new URL(groupFeed.getLink(Link.Rel.NEXT,
    Link.Type.ATOM).getHref()), GsaFeed.class);
}
```

To retrieve a single group:

```
GsaEntry groupEntry = service.getEntry(new URL("http://Search_Appliance:8000/a/
feeds/group/2.0/domain/testGroup"),
  GsaEntry.class);
System.out.println("Group Name: " + groupEntry.getProperty("groupName"));
```

With search appliance software release 7.0, groups now can have several additional attributes: Domain, Namespace, and Case Sensitivity. Here is an example in retrieving those fields.

```
GsaEntry groupEntry = service.getEntry(new URL("http://Search_Appliance:8000/a
/feeds/group/2.0/domain/testGroup/namespace/Default/domain/My_domain/caseType
/everything-case-sensitive"),
 GsaEntry.class);
System.out.println("Group Name: " + groupEntry.getProperty("groupName"));
System.out.println("Group Protocol Buffer: " +
groupEntry.getGsaContent("groupProto"));
```

Deleting Members of a Group

To delete members of a group, use the following `DELETE` request:

```
DELETE http://Search_Appliance:8000/a/feeds/group/2.0/domain/groupId
```

Take note that this request only deletes members of a group, it does not delete the empty group.

Adding a Member to a Group

To add a new member to a group:

```
GsaEntry memberEntry = new GsaEntry();
memberEntry.addProperty("memberId", "john");
memberEntry.addProperty("memberType", "user");
// Adds member user "john" to group "testGroup"
service.insert(new URL("http://Search_Appliance:8000/a/feeds/group/2.0/domain/" +
  "testGroup" + "/member"), memberEntry);
```

In release 7.0, to add users with the extra attributes of Domain, NameSpace, and Case Sensitivity:

```
GsaEntry memberEntry = new GsaEntry();
memberEntry.addProperty("memberId", "john");
memberEntry.addProperty("memberType", "user");
memberEntry.addProperty(“memberNamespaceId”, “Default”);
memberEntry.addProperty(“memberDomainId”, “My_domain”);
memberEntry.addProperty(“memberCaseType”, “everything-case-sensitive”);
// Adds member user "john" to group "testGroup"
service.insert(new URL("http://Search_Appliance:8000/a/feeds/group/2.0/domain/" +
 "testGroup" + "/member"), memberEntry);
```

Retrieving Group Members

To retrieve all members of a group:

```
GsaFeed memberFeed = service.getFeed(new URL("http://Search_Appliance:8000/a
/feeds/group/2.0/domain/" + "testGroup" + "/member"), GsaFeed.class);
for(GsaEntry memberEntry : memberFeed.getEntries()) {
  System.out.println("Member Name: " + memberEntry.getProperty("memberId"));
  System.out.println("Member Type: " + memberEntry.getProperty("memberType"));
}
```

If the number of members in a group is more than 500, the result is represented in multiple pages—you can access the next page as follows:

```
if (memberFeed.getLink(Link.Rel.NEXT, Link.Type.ATOM) != null) {
  memberFeed = service.getFeed(new URL(memberFeed.getLink(Link.Rel.NEXT,
Link.Type.ATOM).getHref()), GsaFeed.class);
}
```

To retrieve a single member of a group:

```
memberEntry = service.getEntry(new URL("http://Search_Appliance:8000/a/feeds
/group/2.0/domain/testGroup/member/" + "john"), GsaEntry.class);
System.out.println("Member Name: " + memberEntry.getProperty("memberId"));
System.out.println("Member Type: " + memberEntry.getProperty("memberType"));
```

To retrieve a single member of a group fully specifying the new fields in release 7.0:

```
memberEntry = service.getEntry(new URL("http://Search_Appliance:8000/a/feeds
/group/2.0/domain/testGroup/namespace/Default/domain/My_domain/caseType
/everthing-case-sensitive/member/" + "john" + “/memberNamespace/Default
/memberDomain/Default/memberCaseType/everthing-case-sensitive), GsaEntry.class);
System.out.println("Member Name: " + memberEntry.getProperty("memberId"));
System.out.println("Member Protocol Buffer: " +
memberEntry.getProperty("memberProto"));
```

Removing a Member From a Group

To remove a member from a group:

```
service.delete(new URL("http://Search_Appliance:8000/a/feeds/group/2.0/domain
/testGroup" + "/member/" + "john"));
```

To remove a fully-specified member from a fully-specified group:

```
service.delete(new URL(
"http://Search_Appliance:8000/a/feeds/group/2.0/domain/testGroup/namespace
/Default/domain/My_domain/caseType/everthing-case-sensitive/member/" + "john"
+ "/memberNamespace/Default/memberDomain/Default/memberCaseType
/everthing-case-sensitive));
```

Policy ACL API Developer's Guide: .NET

The sections that follow specify policy ACL rules using .NET.

Getting Started

The google-enterprise-gdata-api (http://code.google.com/p/google-enterprise-gdata-api) open source site provides ZIP files that contain sample C#.NET example files, the .NET client library (DLLs), source code, and a sample application for your reference.

The information in this section helps you understand how to write your own applications based on the C#.NET client library and how to run the provided open source sample applications.

You can use the open source sample C# files (http://code.google.com/p/google-enterprise-gdata-api/source/browse/#svn/trunk/cs/sample) as models for your own development.

Before starting, you need the following:

- Microsoft Visual C# 2008 Express Edition (http://www.microsoft.com/exPress/download/#webInstall), which includes a free version of Visual Studio so that you can work with the .NET client library.
- The API software in the gsa-admin-api-cs-1.0.0.zip ZIP file (http://google-enterprise-gdata-api.googlecode.com/files/gsa-admin-api-cs-1.0.0.zip), which contains the client library and the sample application files.
- Admin Console user name and password for the search appliance to which you direct your commands.

Getting Samples

After you download the software and acquire search appliance credentials, get started as follows:

1. Unzip the API ZIP file and navigate to the `cs` folder. The client library DLL files are present in the `lib` folder and the sample application is present in the `sample` folder.
2. Start Microsoft Visual C# 2008 Express Edition and click **File** > **Open Project**, browse to the location where you stored the `gsa.sln` solution file, and open the solution file, which appears in the Solution Explorer.

3. Click **Build** > **Build Solution** to build the project. Ensure that the build runs without errors. The binaries and DLL files are put in the `cs\sample\bin\Release` folder. The output binary is the `GsaCommandLine.exe` executable file in the `Release` folder.

4. Open a command prompt and run the command to view its options:

```
C:\GoogleDataAdministrativeAPI\cs\sample\bin\Release>GsaCommandLine.exe
Usage: GsaCommandLine <command> <options> <query_parameters> feed entry
commands:
    retrieve
    update
    insert
    delete
options:
  --protocol:
  --hostname:
  --port:
  --username:
  --password:
  --input: The input entry file for insert/update
<query_parameters>:
  All the query parameters can be specified by --<query>=<value>

Example:
  GsaCommandLine retrieve --protocol=http --hostname=gsa1 --port=8000
    --username=user --password=password config crawlURLs

C:\GoogleDataAdministrativeAPI\API-Gdata\cs\sample\bin\Release>
```

5. You can run a command using the parameters that are shown. Be sure for the host name to use just the search appliance name and not a domain name.

Building Your Applications

This section explains how to build your own applications using the client library outside the solution file provided by the ZIP archive.

To build an application:

1. Copy the following client library DLL files from the `cs\lib` folder to your development folder and add them in the reference path:
   - `Google.GData.Apps.dll`
   - `Google.GData.Client.dll`
   - `Google.GData.Extensions.dll`
   - `Google.GData.Gsa.dll`
2. In Visual Studio, create or open a new project.
3. Click **Project** > **Add Reference**.
4. Click the **Browse** tab and navigate to the folder where you copied the DLL files.
5. Select the DLLs to use in your project.

API Authentication

Before making API calls with the .NET client library, you must construct a new `GsaService` object.

In the constructor that follows, replace *myUserId* and *myPassword* with your Admin Console authentication information:

```
GsaService service = new GsaService(gsaAddr, "myUserId", "myPassword");
```

Pattern ACL API

The code in the sections that follow specifies the URL pattern for a rule.

Creating an ACL Rule

To create an ACL rule:

```
GsaEntry entry = new GsaEntry();
entry.AddGsaContent("urlPattern", "http://example.com");
entry.AddGsaContent("acl", "group:testGroup user:john");
service.InsertEntry("policyAcls", entry);
```

Retrieving ACL Rules

To retrieve all ACL rules:

```
GsaFeed feed = service.GetFeed("policyAcls");
```

To retrieve an ACL rule for a URL pattern:

```
GsaEntry entry = service.GetEntry("policyAcls", "http://example.com");
```

Updating an ACL Rule

To update an ACL rule:

```
GsaEntry entry = new GsaEntry();
entry.AddGsaContent("urlPattern", "http://abc2.com");
entry.AddGsaContent("acl", "group:testGroup user:john");
service.UpdateEntry("policyAcls", "http://example.com", entry);
```

Deleting an ACL Rule

To delete an ACL rule:

```
service.DeleteEntry("policyAcls", "http://example.com");
```

Group and Member API

The code in the sections that follow specifies which users and groups are allowed to access a URL pattern that you set in the pattern ACL API.

Creating a Group

To create a new group:

```
GsaEntry insertEntry = new GsaEntry();
insertEntry.Properties.Add(new PropertyElement("groupId", "testGroup"));
service.Insert(new Uri("http://Search_Appliance:8000/a/feeds/group/2.0/domain"),
insertEntry);
```

Retrieving Groups

To retrieve all groups:

```
GsaFeed resultFeed = service.Query(new FeedQuery("http://gsa.example.com:8000/a/
feeds/group/2.0/domain")) as GsaFeed
```

If the number of groups in the search appliance is more than 500, the result is represented in multiple pages—you can access the next page as follows:

```
if ((next = feed.Links.FindService("next", null)) != null)
{
  resultFeed = Query(new Uri(next.HRef.ToString())) as GsaFeed;
}
```

Deleting Members of a Group

To delete members of a group, use the following `DELETE` request:

```
DELETE http://Search_Appliance:8000/a/feeds/group/2.0/domain/groupId
```

Take note that this request only deletes members of a group, it does not delete the empty group.

Adding a Member to a Group

To add a member to a group:

```
GsaEntry insertEntry = new GsaEntry();
insertEntry.Properties.Add(new PropertyElement("memberId", "john"));
insertEntry.Properties.Add(new PropertyElement("memberType", "User"));
// Adds member user "john" to group "testGroup"
service.Insert(new Uri("http://Search_Appliance:8000/a/feeds/group/2.0/domain
/testGroup/member"), insertEntry);
```

Retrieving Group Members

To retrieve all members of group:

```
GsaFeed resultFeed = service.Query(new FeedQuery("http://Search_Appliance:8000/a
/feeds/group/2.0/domain/testGroup/member")) as GsaFeed
```

If the number of members in a group is more than 500, the result is represented in multiple pages—you can access the next page as follows:

```
if ((next = feed.Links.FindService("next", null)) != null)
{
  memberFeed = Query(new Uri(next.HRef.ToString())) as GsaFeed;
}
```

Removing a Member From a Group

To remove a member from a group:

```
service.Delete("http://Search_Appliance:8000/a/feeds/group/2.0/domain/testGroup
/member/john");
```

Policy ACL API Developer’s Guide: Protocol

The sections that follow provide an introduction to the policy ACL protocol. See also the “API Operations” and “XML Element Definitions” sections in the *Administrative API Developer’s Guide: Protocol*.

API Authentication

You can send API requests over HTTPS or HTTP. To use this API, you need to specify an authentication token with each API request. The search appliance uses the token to authorize access to the operation that you request. Authentication tokens are available only to users who have administrative rights to the search appliance, and the tokens authorize operations only within a search appliance.

To obtain an authentication token, submit an HTTPS `POST` request structured as form post to the following URL:

```
https://Search_Appliance:8443/accounts/ClientLogin
```

The following guidelines apply to the request:

- Include in the `POST` body the following parameters:
  - Email—user name for an Admin Console administrator account.
  - Passwd—password for the Admin Console account. The user name and password values must be URL-encoded. For example, the URL-encoded form of the `AcQ.87@` password is the `AcQ%2E87%40` value.
- The `POST` request must specify the value `application/x-www-form-urlencoded` for the `Content-Type` header.

The search appliance returns a response that contains your authentication token in response to the `POST` request. The authentication token is the `Auth` value on that page, and you need to extract the token from the page. When you submit an API request, you must set the `Content-Type` and authorization headers as follows:

```
Content-type: application/atom+xml
Authorization: GoogleLogin auth=your-authentication-token
```

Note: Authentication tokens expire after 24 hours or 30 minutes when not in use. Submit a request to the URL at least once again. We recommend that you keep the token in memory rather than writing the token to a file.

Pattern ACL API

Create, retrieve, update, and delete ACL rules for URL pattern on a search appliance.

A set of ACL rules can be specified for a URL pattern. The following parameters are used in the `name=` attribute:

Parameters	Description
`urlPattern`	The URL pattern for which the ACL rules apply.
`acl`	The ACLs. The following example shows the format of the ACLs: `group:engineer user:polly user:ji`

`aclProto`	The ACLs in protocol buffer format. The following example shows the format of the ACL:
	`entries <`
	`  gsa_entry <`
	`    access: 1`
	`    principal <`
	`      scope: 2`
	`      name: "testGroup"`
	`      name_space: "Default"`
	`      case_sensitive: 0`
	`    >`
	`  >`
	`>`

`protoAcls`	Set to `true` if `acl` is in protocol buffer format.

The following are the properties:

Property	Description
`query`	A query string to perform a URL pattern search. The matched ACL rules should contain a URL pattern and the matching mode, which depends on the `matchMode` parameter.
`matchMode`	The matching mode for the URL patterns. The possible values are: • `all`—Match any rules. • `url`—The input query is a URL and only the ACL rules could apply to the URL that is returned, for example, the rule with URL pattern "example.com" matches the "http://example.com/test/index.html" input query. • `document`—Only return document-level ACL rules. For example, the rule with the URL pattern "example.doc$" matches the "example" input query. • `coarseGrain`—Only return non-document-level ACL rules. For example, the rule with URL pattern "example.com" matches the input query "example" but the rule with URL pattern "example.doc$" does not.
`startLine`	The starting line number of a result, the default value is `0` results.
`maxLines`	The number of result lines in a response, the default value is `100` lines of results.

Creating an ACL Rule

To create an ACL rule, send an authenticated `POST` request to the following URL:

```
http://Search_Appliance:8000/feeds/policyAcls
```

To create a new ACL rule with a default setting, use the following entry:

```
<?xml version='1.0' encoding='UTF-8'?>
<entry xmlns='http://www.w3.org/2005/Atom'
  xmlns:gsa='http://schemas.google.com/gsa/2007'>
  <gsa:content name='urlPattern'>http://example.com</gsa:content>
  <gsa:content name='acl'>user:john group:eng</gsa:content>
</entry>
```

Retrieving ACL Rules

To retrieve a list of ACL rules, send an authenticated `GET` request to the following URL:

```
http://Search_Appliance:8000/feeds/
policyAcls[?[query=][matchMode=][startLine=][maxLines=]]
```

The following example shows a sample result:

```
<?xml version='1.0' encoding='UTF-8'?>
<feed xmlns='http://www.w3.org/2005/Atom'
  xmlns:openSearch='http://a9.com/-/spec/opensearchrss/1.0/'
  xmlns:gsa='http://schemas.google.com/gsa/2007'>
  <id>http://gsa.example.com:8000/feeds/policyAcls</id>
  <updated>2009-04-27T12:57:56.152Z</updated>
  <link rel='http://schemas.google.com/g/2005#feed' type='application/atom+xml'
    href='http://gsa.example.com:8000/feeds/policyAcls'/>
  <link rel='self' type='application/atom+xml'
    href='http://gsa.example.com:8000/feeds/policyAcls'/>
  <generator version='0.5' uri='http://gsa.example.com:8000/gsa'>Google Search
Appliance</generator>
  <openSearch:startIndex>1</openSearch:startIndex>
  <entry>
    <id>http://gsa.example.com:8000/feeds/policyAcls/example.com</id>
    <gsa:content name='entryID'>example.com</gsa:content>
    <gsa:content name='urlPattern'>example.com</gsa:content>
    <gsa:content name='acl'>group:eng user:john</gsa:content>
  </entry>
  ...
</feed>
```

To retrieve an ACL rule for a URL pattern, send an authenticated `GET` request to the following URL:

```
http://Search_Appliance:8000/feeds/policyAcls/Url_Pattern
```

The following example shows a sample result:

```
<?xml version='1.0' encoding='UTF-8'?>
<entry>
  <id>http://gsa.example.com:8000/feeds/policyAcls/http%3A%2F%2Fexample.com</id>
  <gsa:content name='entryID'>http%3A%2F%2Fexample.com</gsa:content>
  <gsa:content name='urlPattern'>http://example.com</gsa:content>
  <gsa:content name='acl'>group:eng user:john</gsa:content>
</entry>
```

Updating an ACL Rule

To update an attribute in an ACL rule for a URL pattern, send an authenticated PUT request to the following URL:

```
http://Search_Appliance:8000/feeds/policyAcls/Url_Pattern
```

The following example entry updates the ACL rule:

```
<?xml version='1.0' encoding='UTF-8'?>
<entry xmlns='http://www.w3.org/2005/Atom' xmlns:gsa='http://schemas.google.com/
gsa/2007'>
  <gsa:content name='urlPattern'>http://example.com</gsa:content>
  <gsa:content name='acl'>user:john group:eng</gsa:content>
</entry>
```

Deleting an ACL Rule

To delete an ACL rule from a search appliance, send an authenticated DELETE request to the following URL:

```
http://Search_Appliance:8000/feeds/policyAcls/Url_Pattern
```

Group and Member API

The code in the sections that follow specifies which users and groups can access a URL pattern for the ACL rule.

Creating a Group

To create a group, use the following POST request:

```
POST http://Search_Appliance:8000/a/feeds/group/2.0/domain
```

Retrieving Groups

To retrieve all groups in a particular domain, use the following GET request:

```
GET http://Search_Appliance:8000/a/feeds/group/2.0/domain[?[start-index=]]
```

Deleting Members of a Group

To delete members of a group, use the following DELETE request:

```
DELETE http://Search_Appliance:8000/a/feeds/group/2.0/domain/groupId
```

Take note that this request only deletes members of a group, it does not delete the empty group.

Sample GroupEntry Request

The following XML sample shows a sample request to create a group. The sample uses the `groupName` to specify the name of the group. For the search appliance, the `groupName` and `groupId` are the same.

```
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:apps="http://schemas.google.com/apps/2006"
  xmlns:gd="http://schemas.google.com/g/2005">
  <apps:property name="groupId" value="us-sales"></apps:property>
  <apps:property name="groupName" value="us-sales"></apps:property>
  <apps:property name="description" value=""></apps:property>
  <apps:property name="emailPermission" value=""></apps:property>
</atom:entry>
```

A newly created group does not have subscribers. The `emailPermission` and `description` properties are not supported by the search appliance.

Sample GroupEntry Response

When you submit a request to create, retrieve, or update a group, the Provisioning API returns an XML response that identifies the group.

The XML code that follows shows a sample API response for a request to create a group.

```
<atom:entry>
  <atom:id>http://gsa.example.com:8000/a/feeds/group/2.0/domain/us-sales</
atom:id>
  <atom:link rel="self" type="application/atom+xml"
    href="http://gsa.example.com:8000/a/feeds/group/2.0/domain/us-sales"/>
  <atom:link rel="edit" type="application/atom+xml"
    href="http://gsa.example.com:8000/a/feeds/group/2.0/domain/us-sales"/>
  <apps:property name="groupId" value="us-sales"></apps:property>
  <apps:property name="groupName" value="us-sales"></apps:property>
  <apps:property name="description" value=""></apps:property>
  <apps:property name="emailPermission" value=""></apps:property>
</atom:entry>
```

Sample GroupFeed Response

When you submit a request to retrieve all groups for a domain or all groups to which a particular user subscribes, the Provisioning API returns an Atom XML feed containing a list of groups, each of which is identified in an <atom:entry> XML block the *Administrative API Developer’s Guide: Protocol*.

The XML code that follows shows a sample API response for a request to retrieve all groups for a domain. Because the `emailPermission` and `description` properties are not supported by the search appliance, the values are specified as an empty string (`""`).

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:apps="http://schemas.google.com/apps/2006"
  xmlns:openSearch="http://a9.com/-/spec/opensearchrss/1.0/">
  <atom:id>http://gsa.example.com:8000/a/feeds/group/2.0/domain</atom:id>
  <atom:updated>2008-12-03T16:33:05.260Z</atom:updated>
  <atom:link href="http://gsa.example.com:8000/a/feeds/group/2.0/domain"
    type="application/atom+xml" rel="http://schemas.google.com/g/2005#feed"></
atom:link>
  <atom:link href="http://gsa.example.com:8000/a/feeds/group/2.0/domain"
    type="application/atom+xml" rel="http://schemas.google.com/g/2005#post"></
atom:link>
  <atom:link href="http://gsa.example.com:8000/a/feeds/group/2.0/domain"
    type="application/atom+xml" rel="self"></atom:link>
  <openSearch:startIndex>1</openSearch:startIndex>
  <atom:entry>
    <id>http://gsa.example.com:8000/a/feeds/group/2.0/domain/us-sales%40domain</
id>
    <atom:updated>2008-12-03T16:33:05.261Z</atom:updated>
    <atom:link href="http://gsa.example.com:8000/a/feeds/group/2.0/domain/us-
sales%40domain"
      type="application/atom+xml" rel="self"></atom:link>
    <atom:link href="http://gsa.example.com:8000/a/feeds/group/2.0/domain/us-
sales%40domain"
      type="application/atom+xml" rel="edit"></atom:link>
    <apps:property name="groupId" value="us-sales"></apps:property>
    <apps:property name="groupName" value="us-sales"></apps:property>
    <apps:property name="emailPermission" value=""></apps:property>
    <apps:property name="description" value=""></apps:property>
  </atom:entry>
  <atom:entry>
    <atom:id>http://gsa.example.com:8000/a/feeds/group/2.0/domain/Staff-
2435%40domain</atom:id>
    <atom:updated>2008-12-03T16:33:05.260Z</atom:updated>
    <atom:link href="http://gsa.example.com:8000/a/feeds/group/2.0/domain/Staff-
2435%40domain"
      type="application/atom+xml" rel="self"></atom:link>
    <atom:link href="http://gsa.example.com:8000/a/feeds/group/2.0/domain/Staff-
2435%40domain"
      type="application/atom+xml" rel="edit"></atom:link>
    <apps:property name="groupId" value="Staff-2435@domain"</apps:property>
    <apps:property name="groupName" value="Staff-2435@domain"</apps:property>
    <apps:property name="emailPermission" value=""</apps:property>
    <apps:property name="description" value=""></apps:property>
  </atom:entry>
  <atom:entry>
  ...
  </atom:entry>
</atom:feed>
```

Adding a Member to a Group

To add a member to a group, use the following `POST` request:

```
POST http://Search_Appliance:8000/a/feeds/group/2.0/domain/groupId/member
```

Retrieving Group Members

To retrieve all members of a group, use the following `GET` request:

```
GET http://Search_Appliance:8000/a/feeds/group/2.0/domain/groupId/
member[?[start-index=]]
```

To retrieve a particular member of a group, use the following `GET` request:

```
GET http://Search_Appliance:8000/a/feeds/group/2.0/domain/groupId/member/
memberId
```

Removing Members From a Group

To remove a group member, use the following `DELETE` request:

```
DELETE http://Search_Appliance:8000/a/feeds/group/2.0/domain/groupId/member/
memberId
```

Sample MemberEntry Request

The following XML shows a sample request to add a member to a group. The XML uses the `memberId` property to specify the member and the `memberType` property to specify the type of member, which can be `user` or `group`.

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:apps="http://schemas.google.com/apps/2006"
  xmlns:gd="http://schemas.google.com/g/2005">
  <apps:property name="memberId" value="susanjones@example.com"/>
  <apps:property name="memberType" value="user"/>
</atom:entry>
```

If `memberType` is not specified and the member being added already exists as a group, then the member will be added as a group member, as in the following request:

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:entry xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:apps="http://schemas.google.com/apps/2006"
  xmlns:gd=’http://schemas.google.com/g/2005">
  <apps:property name="memberId" value="us-sales@example.com"/>
</atom:entry>
```

Here `us-sales@example.com` already exists as a group.

Sample MemberEntry Response

When you submit a request to add a member to a group, the Group API returns an XML response that identifies the newly added member. Following a request to add a member to a group, this object does not serve any purpose except to confirm that the request was successful.

The XML that follows shows a sample API response for a request that retrieves a specific member in a group.

```
<atom:entry>
  <atom:id>http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/suejones%40example.com</atom:id>
  <atom:link rel="self" type="application/atom+xml"
    href="http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/suejones%40example.com"/>
  <atom:link rel="edit" type="application/atom+xml"
    href="http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/suejones%40example.com"/>
  <apps:property name="memberId" value="suejones@example.com"/>
  <apps:property name="memberType" value="User"/>
  <apps:property name="directMember" value="true"/>
</atom:entry>
```

Sample MemberFeed Response

When you submit a request to retrieve all members for a group, the Group API returns an Atom XML feed identifying a list of member, each of which is identified in an `<atom:entry>` XML block.

The XML code that follows shows a sample API response for a request to retrieve all members of a group.

```
<?xml version="1.0" encoding="UTF-8"?>
<atom:feed xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:apps="http://schemas.google.com/apps/2006"
  xmlns:openSearch="http://a9.com/-/spec/opensearchrss/1.0/">
 <atom:id>http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member</atom:id>
 <atom:link rel="http://schemas.google.com/g/2005#feed" type="application/
atom+xml"
   href="http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member"/>
 <openSearch:startIndex>1</openSearch:startIndex>
 <atom:entry>
  <atom:id>http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/suejones%40example.com</atom:id>
  <atom:link rel="self" type="application/atom+xml"
    href="http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/suejones%40example.com"/>
  <atom:link rel="edit" type="application/atom+xml"
    href="http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/suejones%40example.com"/>
  <apps:property name="memberId" value="suejones@example.com"/>
  <apps:property name="memberType" value="User"/>
  <apps:property name="directMember" value="true"/>
 </atom:entry>
 <atom:entry>
   <atom:id>http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/ca-sales%40example.com</atom:id>
   <atom:link rel="self" type="application/atom+xml
     href="http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/ca-sales%40example.com"/>
   <atom:link rel="edit" type="application/atom+xml
     href="http://gsa.example.com:8000/a/feeds/group/2.0/example.com/us-sales/
member/ca-sales%40example.com"/>
   <apps:property name="memberId" value="ca-sales@example.com"/>
   <apps:property name="memberType" value="Group"/>
   <apps:property name="directMember" value="true"/>
 </atom:entry>
 <atom:entry>
   ...more entries...
 </atom:entry>
</atom:feed>
```