

Network Working Group  
Internet Draft: ACAP Authid Dataset Classes

S. Hole  
A. Melnikov  
ACI WorldWide/MessagingDirect  
June 2002  
Expires in 6 months

Document: [draft-ietf-acap-authid-03.txt](#)

## ACAP Authorization Identifier Dataset Classes

### Status of this memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at  
<<http://www.ietf.org/ietf/lid-abstracts.txt>>

The list of Internet-Draft Shadow Directories can be accessed at  
<<http://www.ietf.org/shadow.html>>.

A version of this draft document is intended for submission to the RFC editor as a Proposed Standard for the Internet Community. Discussion and suggestions for improvement are requested. This document will expire six months after publication. Distribution of this draft is unlimited.

### Copyright Notice

Copyright (C) The Internet Society 2002. All Rights Reserved.

### [0.](#) Administrivia

#### [0.1.](#) Open Issues

Internet Draft

ACAP Authid Dataset Classes

June 2002

- 1) The calculation for resolving to a unique list of users from a set of group references might need some discussion.
- 2) There is an inconsistency between `userid.memberof` and the group entry syntax. `userid.memberof` uses `relativeURL` syntax, that coincides with `entry-path` syntax. It is desirable to use the same syntax for `groupid-entry-ref/userid-entry-ref`. However the syntax for "entry" attribute doesn't allow slashes, so `groupid-entry-ref` and `userid-entry-ref` use "groupid." and "userid." as prefixes. Suggestions are welcome.

#### [0.2.](#) Changes from revision 00

- 1) Added a glossary of terms section at the beginning.
- 2) Changed the `group.members` attribute of a `groupid` entry from a multivalued attribute to a subdataset attribute. This is to address the scaling issues of very large groups, insertion and deletion. Each entry in the subdataset refers to a member -- either a `userid` or `groupid`. The namespace for `userid` and `groupid` is separated by prefixing all `userid` member references with the "userid." prefix string, and all `groupid` member references are prefixed with a "groupid." string.
- 3) Punted on the issue of recursive or circular group references. It is up to the agent using the group information to do spanning tree calculations and/or set reductions to arrive at a unique membership list. This MAY include the ACAP server itself if it uses the authid dataset classes for authorization information.

#### [0.3.](#) Changes from revision 01

- 1) Some attributes were missing text whether they are single valued or multi valued

- 2) Fixed typo in ACL definition. Fixed attribute names to conform to ACAP spec.

- 3) Corrected userid.whitepage-info definition - relativeURL was used instead of URL
- 4) Added text explaining that fully qualified userids/groupids must be used in ACLs.
- 5) Added example
- 6) Added capability registration forms. Updated address and copyright information. Some other minor editorial changes.

## 1. Introduction

Most distributed (client/server) applications require an authentication process between the client and the server before the server will grant access to its managed resources. Many applications provide varying levels of access to server resources based on a combination of authentication credentials and access control rules. The collection of information used to control access to resources is called "authorization information".

The authorization identifier datasets contain lists of users and groups of users that can be used by applications for authorization purposes. Access control mechanisms can be abstracted from underlying authentication mechanisms and credential formats. They can be extended to include group memberships in dynamic calculations for access rights to resources or in definition of one time authorization certificates.

The Application Configuration Access Protocol (ACAP) supports the remote storage and access of many types of structured configuration information. The authorization identifier datasets specification describes the "userid" and "groupid" datasets which contain the authorization information. It also describes ACAP server capabilities that advertise a server's support for authorization user and group semantics.

## [2.](#) Conventions used in this document

Hole, Melnikov

[Page 3]

---

Internet Draft

ACAP Authid Dataset Classes

June 2002

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document are to be interpreted as described in [[KEYWORD](#)].

The attribute syntax specifications use the Augmented Backus-Naur Form (ABNF) notation as specified in [[ABNF](#)].

## [3.](#) Definition of Terms

Historically, different operating and network systems have used authentication and authorization terminology interchangeably. They often did not distinguish between authentication and authorization, binding both into a single process. Terms like "userid", "username", "login", "access" and others are used and mean somewhat different things in different systems.

Following is an introductory glossary of the terminology used by this specification. The terminology is defined specifically for Internet client/server applications, although the terminology could be applied to any application. It reuses some historical terms, but defines each with a specific role, scope and usage.

### [3.1.](#) Authentication and authorization model

Granting access to server resources in a client/server application is a two step process. Step 1 is called "authentication", Step 2 is "authorization". Step 1 is performed once per session and establishes the identity of the individual that desires to access

server resources. Step 2 may be executed many times as access rights for the authenticated client are calculated for different resources managed by the server.

In the ACAP model, authentication information is held independently of and bound to authorization information using the ACAP authorization identifiers.

### [3.2.](#) Glossary of Terms

#### authid

The "authentication identifier" used to uniquely identify the individual connecting to a service. The syntax and semantics of authids are specific to a particular authentication mechanism, network, and/or host operating system.

#### userid

The "authorization user identifier" used to identify an individual that a service will grant access rights to for its managed resources. The syntax and semantics of ACAP userids are application independent.

#### groupid

The "authorization group identifier" used to identify a set of individuals and/or groups that a service will grant access rights to for its managed resources. The syntax and semantics of ACAP groupids are application independent.

#### rights

The type of access that an individual is granted to a resource. The specific rights that can be granted to a resource is applica-

tion specific.

## ACL

An "access control list" is a set of rules that binds an authorization id (userid, groupid) to a set of rights. The form and content of ACLs are application specific.

## authentication

The negotiation between a client and server application that unambiguously establishes the identity of the client and server parties.

## authorization

The calculation performed by the server to grant access rights to a resource for an authenticated client.

### [4.](#) Authorization user identifiers

An ACAP "userid" (user identifier) is an abstraction for an individual user that accesses server resources -- an authorization user. Typically, this is a person acting as him or herself, a person acting in a role, or an application process. Access rights to server resources can be granted or denied to a userid.

Authentication information is tied to a userid by an authentication mechanism specific "authid" (authentication identifier). More than one authid can be associated with a single userid.

Userids can be listed and displayed by a client without giving away critical information on authentication information -- specifically lists of authids. Using ACAP access control lists, the authids

tied to a userid MAY be searched by a client but SHOULD NOT be retrievable by a client.

#### [4.1.](#) ACAP userid dataset class

Datasets whose names begin with "/userid/" contain "userid" entries as defined in this specification. If present, an ACAP server SHOULD calculate access rights for its own information resources using the authorization information in this dataset.

#### [4.2.](#) Userid entry attributes

A "userid" entry defines an authorization user for an application. It is used by the application to grant or deny access to application resources. An application supporting ACAP "USER" authorization semantics (as defined in [section 5.](#)) binds userids to its resource access control rules. Resource access rights are calculated by applying, in an application specific way, the access control rules that are bound to the current user's userid.

##### [4.2.1.](#) Mandatory userid attributes

###### entry

The "entry" attribute MUST be defined for a userid entry. Its value is a userid. The "entry" attribute is used by applications to calculate access rights to server resources. This SHOULD include the ACAP server itself. The syntax for the "entry" attribute is defined in [ACAP].

###### userid.authid

The "userid.authid" attribute MUST be defined for userid entry. It MAY be multivalued. It contains a list of authentication mechanism specific authentication identifiers that bind to this userid.

```
userid.authid ::= 1*TEXT_UTF8_CHAR
                ;; multi-valued
```

#### [4.2.2.](#) Optional userid attributes

##### userid.displayname

The "userid.displayname" attribute MAY be defined for a userid entry. It MUST be single valued. It contains a name string which is suitable for presentation by an ACAP client. If present in a userid entry, clients SHOULD present this value to the user rather than the value of the "entry" attribute. It is assumed to contain a more descriptive label for the user than the userid itself, e.g. the user's full name.

userid.displayname ::= 1\*TEXT\_UTF8\_CHAR

##### userid.description

The "userid.description" attribute MAY be defined for a userid entry. It MUST be single valued. The value contains text that provides an extended description of the user. This information can be presented to a user to assist them in disambiguating userid entries with similar (or identical) "userid.displayname" attribute values.

userid.description ::= 1\*TEXT\_UTF8\_CHAR

##### userid.whitepage-info

The "userid.whitepage-info" attribute MAY be defined for a userid entry. It MAY be multivalued. The value contains one or more URL's that reference whitepages information for the user. There are no restrictions on the type of the URL, but it is most likely that the URL will be either an ACAP URL pointing to an addressbook dataset class entry or an LDAP URL pointing to a person entry. This information can be presented to a user to assist them in disambiguating userid entries with similar (or identical) "userid.displayname" attribute values.

userid.whitepage-info ::= URL



```
;; as defined in [REL-URL]
;; ACAP relative URL is defined in [ACAP]
```

userid.memberof

The "userid.memberof" attribute MUST be defined for an entry if the server supports the ACAP "GROUP" authorization semantics. It MAY be multivalued. The value of the attribute is the list of groupids that the userid is a member of. It is provided as an optimization convenience to the client in the presence of group authorization semantics as defined in [section 5](#). The value is readonly and MUST be calculated by the server.

```
userid.memberof ::= relativeURL
                    ;; as defined in [REL-URL]
                    ;; ACAP relative URL is defined in [ACAP]
```

## [5](#). Authorization group identifiers

An ACAP "groupid" (group identifier) is an abstraction for a set of users that access server resources -- an authorization group. A groupid entry contains a list of userids that are members of the group. Access rights to server resources can be granted or denied to a groupid.

### [5.1](#). ACAP groupid dataset class

Datasets whose names begin with "/groupid/" are assumed to contain groupid entries as defined in this specification. If present, an ACAP server SHOULD support group authorization semantics defined in [section 5](#).

### [5.2](#). Groupid entry attributes

A "groupid" entry defines an authorization group for an application. It is used by an application to grant or deny access to application resources. An application supporting ACAP "GROUP" authorization semantics (as defined in [section 5](#).) binds groupids to its resource access control rules. Resource access rights are calculated by applying, in an application specific way, the access control rules that are bound to the groupids that the current user's userid is a member of.

Each "groupid" entry is a subdataset entry. Each entry in the

subdataset is called a "member" and references either a userid or another groupid entry.

#### [5.2.1.](#) Mandatory groupid attributes

##### entry

The "entry" attribute MUST be defined for a groupid entry. Its value is a groupid and it is used by applications to calculate access rights to server resources. This SHOULD include the ACAP server itself. The syntax for the "entry" attribute is defined in [ACAP].

##### subdataset

The "subdataset" attribute MUST be defined for a groupid entry. Its value is a relative URL pointing to a dataset whose entries are the members of the group. The syntax for the "subdataset" attribute is defined in [ACAP].

#### [5.2.2.](#) Optional groupid attributes

##### groupid.name

The "groupid.name" attribute MAY be defined for a groupid entry. It MUST be single valued. It contains a name string which is suitable for presentation by an ACAP client. If present in a groupid entry, clients SHOULD present this value to the user rather than the value of the "entry" attribute. It is assumed to contain a more descriptive label for the group than the groupid itself, e.g. the group's organizational title.

groupid.name ::= 1\*TEXT\_UTF8\_CHAR

##### groupid.description

The "groupid.description" attribute MAY be defined for a groupid entry. It MUST be single valued. The value contains text that describes the group.

```
groupid.description ::= 1*TEXT_UTF8_CHAR
```

### [5.3.](#) Group member entry attributes

An ACAP group is a subdataset whose entries enumerate the membership of the group. Each entry references a user entry in the "/userid" dataset class, or a group entry in the "/groupid" dataset class.

#### [5.3.1.](#) Mandatory member entry attributes

##### entry

The "entry" attribute MUST be defined for a member entry. In addition to the restrictions placed on "entry" by [ACAP], a member entry is constrained to be of the form "userid.<userid>" or "groupid.<groupid>", where "<userid>" is a userid entry defined in the "/userid" dataset and "<groupid>" is a groupid entry defined in the "/groupid" dataset. The formal syntax for a member entry name is:

```
groupid.member = userid-entry-ref / groupid-entry-ref
```

```
groupid-entry-ref = "groupid." entry-name
                  ;; entry-name refers to an entry in the
                  ;; "/groupid" dataset
```

```
userid-entry-ref = "userid." entry-name
                  ;; entry-name refers to an entry in the
                  ;; "/userid" dataset
```

Note that there are no restrictions on the membership of a group. In particular a group may include itself as a member. Elimination of recursive references to groups MUST be performed by the agent responsible for calculating group membership attributes like that defined in [section 4.2.2](#), or by agents that use group information

in rights calculations.

## [6.](#) ACAP authorization

The ACAP authorization information can be used by any application, including the ACAP server itself. The following sections describe the use of the authorization identifier datasets by an ACAP application (client or server) itself.

Hole, Melnikov

[Page 10]

---

Internet Draft

ACAP Authid Dataset Classes

June 2002

Other applications are assumed to provide their own definitions for use of ACAP authorization information. Specifically, they are expected to define how they notify clients that their access control mechanisms make use of ACAP authorization information datasets.

### [6.1.](#) Userid access semantics

If an ACAP server supports the `"/userid"` dataset, then it SHOULD use the authorization information provided by it for access control purposes. After successful authentication to the ACAP server, an authorization userid should be selected as the "current user" for the ACAP server, either using the authid mapping information in the userid entries, or using explicit userid information supported by the authentication mechanism. ACAP ACL's are based on userids rather than authids. Resource access rights are calculated relative to the current userid.

#### [6.1.1.](#) ACAP "USER" capability

If an ACAP server supports the `"/userid"` dataset and userid authorization semantics, then it MUST express the "USER" capability in an ACAP capability response. The "USER" capability informs an ACAP client that it MUST use the `"/userid"` dataset contents for any ACL management on the server. Fully qualified userids in the form of `userid-entry-ref` ([Section 5.3.1](#)) MUST be used in ACLs. If a server does not express the "USER" capability, then the client will assume that the server uses authid information in ACL's.

USER capability registration can be found below:

To: iana@iana.org  
Subject: Registration of ACAP capability

Capability name:  
USER

Capability keyword:  
USER

Capability arguments:  
None

Published Specification(s):  
This document

Hole, Melnikov

[Page 11]

---

Internet Draft

ACAP Authid Dataset Classes

June 2002

Person and email address to contact for further information:  
See Authors' Addresses section of this document

## [6.2.](#) Group access semantics

If an ACAP server supports the `"/groupid"` dataset, then it SHOULD use the authentication information in it. ACAP ACL's can include groupids in the ACL. Resource access rights are calculated relative to the current userid, and all groups that the current userid is a member of.

### [6.2.1.](#) ACAP "GROUP" capability

If an ACAP server supports the `"/groupid"` dataset and userid authorization semantics, then it MUST express the "GROUP" capability in an ACAP capability response. The "GROUP" capability informs an ACAP client that it MUST use the `"/groupid"` dataset contents for any ACL management on the server. Fully qualified groupids in the form of `groupid-entry-ref` ([Section 5.3.1](#)) MUST be used in ACLs. If a server does not express the "GROUP" capability, then the client will assume that the server does not support group semantics, and should not present group information in ACAP ACL management functions.

In addition, the server MUST support calculation of the "userid.memberof" attribute in the "/userid" dataset class entries.

GROUP capability registration can be found below:

To: iana@iana.org  
Subject: Registration of ACAP capability

Capability name:  
GROUP

Capability keyword:  
GROUP

Capability arguments:  
None

Published Specification(s):  
This document

Person and email address to contact for further information:  
See Authors' Addresses section of this document

## [7](#). Examples

Server expresses both USER and GROUP capabilities.

"/userid" dataset:

```
entry                = ""
dataset.userid.displayname.acl          =
("groupid.anyonex" "groupid.topxrwia"
"userid.anyoner")           ;; note that refers to the US-
ASCII                       ;; horizontal tab character
dataset.userid.authid.acl    = ()

entry                = "r_migal"
userid.authid        = ("roman" "roman.migal" "migal"
"roman@execmail.com")
userid.displayname    = "Roman Migal"
userid.description    = "Application Developer"
```

```

userid.whitepage-info    = ("/addressbook/user/roman")
userid.memberof          = ("/groupid/anyone")

entry                    = "s_hole"
userid.authid            = ("steve" "steve.hole" "steve@exec-
mail.com")
userid.displayname       = "Steve Hole"
userid.description       = "MessagingDirect CT0"
userid.whitepage-info    = ("/addressbook/user/steve"
"ldap://directory.messagingdi-
rect.com/mail%3Dsteve.hole%40execmail.com")
userid.memberof          = ("/groupid/top" "/groupid/execmail"
"/groupid/devel" "/groupid/anyone")

entry                    = "a_melnikov"
userid.authid            = ("alexey" "alexey.melnikov" "alexey"
"alexey@execmail.com")
userid.displayname       = "Alexey Melnikov"
userid.description       = "Software Developer"
userid.whitepage-info    = ("/addressbook/user/mel"
"http://sites.netscape.net/aamelnikov/index.html"
"ldap://directory.messagingdirect.com/mail%3Dalexey.mel-
nikov%40execmail.com")
userid.memberof          = ("/groupid/devel" "/groupid/store"
"/groupid/anyone")

entry                    = "anonymous"
userid.authid            = ("anonymous")
userid.displayname       = "Anonymous user"
userid.description       = "Used for accessing shared data"

```

```

userid.whitepage-info    = ()
userid.memberof          = ()

```

"/groupid" dataset

```

entry                    = "anyone"
subdataset               = (".")
"acap://other.acap.server.com/groupid/all")
groupid.name              = "All registered users"
groupid.description       = "Note that this entry is different from

```

/userid/anyone"

```
entry          = "devel"  
subdataset     = "."  
groupid.name   = "Software Developers"  
groupid.description = ""
```

"/groupid/anyone" dataset

```
entry          = "userid.r_migal"  
  
entry          = "userid.s_hole"  
  
entry          = "userid.a_melnikov"
```

"/groupid/devel" dataset

```
entry          = "userid.s_hole"  
  
entry          = "userid.a_melnikov"
```

"/groupid/store" dataset

```
entry          = "userid.a_melnikov"
```

"/groupid/top" dataset

```
entry          = "userid.s_hole"
```

"/groupid/execmail" dataset

```
entry          = "userid.s_hole"
```

## 8. References

[ACAP] Newman, C., Myers, J. G., "Application Configuration Access Protocol", [RFC 2244](#), July 1997.



[KEYWORD] S. Bradner, "Key words for use in RFCs to Indicate Requirement Level", [RFC 2119](#), March 1997.

[ABNF] Dave Crocker, P. Overell, "Augmented BNF for Syntax Specifications: ABNF", [RFC 2234](#), July, 1997

[REL-URL] Fielding, "Relative Uniform Resource Locators", [RFC 1808](#), UC Irvine, June 1995.

## [9](#). Security Considerations

This specification defines a protocol for storing, accessing and managing application resource authorization information. It is expected that this information will be used to grant and/or deny access to users and groups for server based resources.

ACAP server access controls should be set correctly on userid entry attributes. Clients SHOULD be able to search for userid entries based on authentication identifier attributes, but SHOULD NOT be able to retrieve any authentication identifier information.

This specification does not define any kind of process, mechanism or protocol for authentication in distributed network applications. Use of the data and protocol elements described in this specification are to be used after successful authentication between the client and server.

This specification does not discuss storage of any kind of authentication credentials, in the form of private keys, shared secrets or passwords in userid entries. The information in the authid dataset is intended purely for authorization and access control purposes.

## [10](#). Acknowledgments

The authors would like to acknowledge the contributions made to this document. Randy Gellens, for editorial comment and feedback on the group membership model. Chris Newman for editorial comment and clarification of the security and access control rights issues for the ACAP server itself.

## [11](#). Authors' Addresses

Steve Hole  
mailto:Steve.Hole@messagingdirect.com

Alexey Melnikov  
mailto:Alexey.Melnikov@messagingdirect.com

ACI WorldWide/MessagingDirect  
900 10117 - Jasper Ave.  
Edmonton, Alberta, T5J 1W8, CANADA

## 12. Full Copyright Statement

Copyright (C) The Internet Society 2002. All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

