

Network Working Group
Internet Draft: ACAP

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

Document: [draft-ietf-acap-option-05.txt](#)

ACAP Application Options Dataset Class

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 an Experimental document. Discussion and suggestions for improvement are requested. This document will expire six months after publication. Distribution of this draft is unlimited.

Abstract

The Application Configuration Access Protocol (ACAP) is designed to support remote storage and access of application option, configuration and preference information. The generalized form of this run-time configuration is called "options". Options consist of any type of structured or unstructured data that an application requires to operate in a user specific manner.

Internet Draft

ACAP Options

December 2002

Storage of application options in an ACAP server substantially solves the "kiosk user" problem for applications -- serial use of a single machine by multiple users. It also solves the "nomadic user" problem -- use of more than one machine on a regular basis by a single user.

Copyright Notice

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

1. Introduction

The Application Configuration Access Protocol (ACAP) is designed to support remote storage and access of application option, configuration and preference information. The generalized form of this run-time configuration is called "options". Options consist of any type of structured or unstructured data that an application requires to operate in a user specific manner.

Storage of application options in an ACAP server substantially solves the "kiosk user" problem for applications -- serial use of a single machine by multiple users. It also solves the "nomadic user" problem -- use of more than one machine on a regular basis by a single user.

The specification defines the "option" dataset class for use with ACAP.

2. Conventions used in this document

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

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

[3.](#) ACAP personal options

[3.1.](#) ACAP option representation

Individual options are represented as ACAP entries in option datasets. The name of the entry, as defined by the "entry" attribute for the entry, is the name of option.

[3.2.](#) Scalar options

Scalar options are ACAP entries that contain simple (unstructured) data. The "option.value" attribute of the entry contains the data for the option. The value can be single or multivalued.

Following is an example of a single and multivalued scalar option:

```
color.preferred
entry = "color.preferred"
  option.value = "blue"
color.list
entry = "color.list"
option.value = ("red" "green" "blue" "cyan" "black")
```

Scalar options that are intended to be used by multiple applications should be registered as defined in [section 4.4](#).

[3.3.](#) Structured options

Structured options are ACAP entries that contain multiple related items of data in a single option. Data for the option is contained in multiple named attributes collectively called "field attributes". Each field attribute can be single or multivalued.

Following is an example of a structured option:

```
color.definition
entry = "color.definition"
option.color.definition.name = "blue"
option.color.definition.rgb = ("blue=255" "red=0" "green=0")
option.color.definition.index = "66"
```

By definition, structured options are application specific. While the content of the field attributes can be obtained by any application, it's intended use is open to interpretation by the application.

[3.4.](#) ACAP option hierarchy

Option sets MAY be represented using ACAP hierarchy. Any entry in an option dataset can also be a hierarchy node by setting the "subdataset" attribute. Applications can model arbitrarily structured configuration information using hierarchical datasets containing scalar or structured options. An option subdataset of scalar options models an associative list. An option subdataset of structured options models an array of structures.

[3.5.](#) ACAP option attribute formal syntax

The naming conventions for option entry attributes is defined by the following ABNF.

```
option-value-attr      = scalar-option-attr / structured-option-attr
scalar-option-attr     = "option.value"
structured-option-attr = "option." name-component
                        1*("." name-component)
```

[4.](#) ACAP option namespace

The general ACAP namespace is organized to promote inheritance between site, host, group, and user specific configuration information, as defined in [\[ACAP\]](#). It defines a "site", "group", "host", and "user" second level hierarchy. The option dataset defines a

third level of hierarchy that promotes inheritance from second level datasets, and isolates user and application specific instances of configuration information.

[4.1.](#) ACAP "/option" dataset class

The dataset class whose name is "/option" is assumed to contain option datasets as defined in this specification.

[4.2.](#) Third level option datasets

Third level option datasets break the option namespace into generic and application specific option sets. This serves two purposes: to promote the creation and inheritance of common options between applications, and isolation of application specific configuration from other applications.

[4.2.1.](#) The "vendor" option namespace

The "vendor" option namespace is a set of datasets, each named in the form "vendor.<vendor-name>", where "<vendor-name>" is the name of a specific application or application vendor.

The entries in the vendor namespace enumerate the applications that make use of ACAP option storage at that level. Each vendor dataset contains option entries and/or subdatasets for a specific vendor applications. All options not in the "standard" namespace MUST be in one of the "vendor" option namespaces.

The "vendor-name" for an application or vendor MUST be registered with IANA according to the rules in [section 5.1](#). The "vendor.x-<vendor-name>" namespace is reserved for temporary use by vendors during product development, or for local applications that are not generally distributed.

Vendors are free to suballocate their namespace as they see fit. For example, a vendor may chose to store options for the various applications that it produces in subdatasets underneath their vendor namespace. For example, the CoolStuff software company may

chose to organize the option namespace for their software products like

```
vendor.CoolStuff.caltool  
vendor.CoolStuff.webtool  
vendor.CoolStuff.mailtool
```

rather than try to register their application names as a vendor-name in the vendor namespace.

[4.2.2](#). The "standard" option namespace

The "standard" option namespace is a dataset named "standard" that contains option entries that are generally applicable to a number of applications. The namespace is reserved for shared options that do not meet the requirements for a separate ACAP dataset class.

The standard option namespace is further partitioned into "standard option classes" where option names are of the form "<standard-option-class-prefix>.<option-name>". Standard option classes are associated with a class of applications, e.g. mail user agents. The "<standard-option-class-prefix>" is a registered string that uniquely identifies the option class, e.g. "mua".

This document defines two standard option class prefixes - "common"

and "X". The "common" standard option class contains options that are generally applicable to a large number of application classes. For example, a default font or window background color. The "X" namespace is reserved for temporary use by software developers during product development.

Following are some examples of the standard option namespace:

```
standard/common.default.font  
standard/common.default.color.background  
standard/X.test-option
```

Option classes and names in the "standard" option namespace MUST be registered with IANA according to the rules in [section 5.2](#). Vendors MUST NOT use the temporary namespace in products that are gen-

erally distributed.

[4.3](#). Example option namespace

The following example demonstrates how the "standard" and "vendor" namespaces isolate application specific and standard configuration within the standard ACAP dataset namespace hierarchy.

```
  /option/  
    site/
```

```

    standard/
    vendor.<vendor-name-1>/
    vendor.<vendor-name-2>/
    ...
    ...
host/
    <host-1>/
        standard/
        vendor.<vendor-name-1>/
        vendor.<vendor-name-2>/
        ...
        ...
    <host-2>/
    ...
    ...
group/
    <group-1>/
        standard/
        vendor.<vendor-name-1>/
        vendor.<vendor-name-2>/
        ...
        ...
    <group-2>/
    ...
    ...
user/
    <user-1>/
        standard/
        vendor.<vendor-name-1>/
        vendor.<vendor-name-2>/
        ...
        ...
    <user-2>/
    ...
    ...

```


The naming conventions for the option dataset are defined by the standard ABNF in [[ACAP](#)] for dataset namespaces, constrained and/or extended by the following ABNF.

```

dname-component  = 1*UTF8-CHAR
                   ;; MUST NOT begin with "." or contain "/"

name-component   = 1*UTF8-CHAR
                   ;; MUST NOT contain ".", "/", "%", or "*"

option-dataset   = "/option/" owner

option-component = standard-component / vendor-component

owner            = owner-site / owner-host / owner-group /
                   owner-user / "~/" option-component

owner-group      = "group/" dname-component "/" option-component

owner-host       = "host/" dname-component "/" option-component

owner-site       = "site/" option-component

owner-user       = "user/" dname-component "/" option-component

standard-component = "standard/" standard-option-class "."
                    name-component
                    ;; The name-component MUST BE an IANA registered
                    ;; option name under the option class.

standard-option-class = "common" / "X" / name-component
                      ;; The name-component MUST BE an IANA registered
                      ;; option class prefix.

vendor-component = vendor-name "/" dname-component

vendor-name      = vendor-token *("." name-component)

vendor-token     = "vendor." name-component
                  ;; The name-component is the name of the
                  ;; application or vendor organization to which
                  ;; the options belong.
```

[5.](#) IANA Considerations

Some portions of the ACAP option namespace are designed to be extensible within the standard. The goals for extensibility include: (1) minimal process to extend the namespace within the standard, (2) promote interoperability within the namespace, and (3) support experimentation and development within the namespace.

This section defines IANA registration templates for standard ACAP options and standard ACAP option classes.

[5.1.](#) Registering vendor names

The "vendor" option namespace is specifically designed to support delegation of authority to individual organizations. The goal is to provide a completely independent namespace for each application vendor. The requirement is that each vendor namespace be uniquely associated with one and only one vendor or product.

The "vendor" option namespace uses the identical registry as the "vendor" dataset class defined in [\[ACAP\]](#). Organizations that wish to register a vendor name for the option namespace MUST follow the procedures outlined in [\[ACAP\]](#).

[5.2.](#) Registering standard options

The "standard" option namespace requires standardization procedures for two different types of object: standard option class prefixes and standard option names.

Standard option class prefixes MUST be registered with IANA. New option class proposals MUST be confirmed using IETF Consensus, primarily through consultation on the ACAP implementors mailing list. Registration of an option class MUST include a review contact that is responsible for approving registrations for option names registered in the option class. Registrations MUST provide the following information when requesting a new standard option class prefix.

To: iana@iana.org

Subject: Registration of ACAP standard option class

Requested standard option class name: <option-class-name>

Requested standard option class prefix: <option-class-prefix>

Requested standard option class short description:

Approval contact: name, e-mail address, address, phone number

Internet Draft

ACAP Options

December 2002

as many contacts as appropriate - must include email address

Standard option names MUST be registered with IANA using the template provided below. All proposals MUST be reviewed and passed by the IANA registered contact for the option class prior to submission. Discussion on new option name proposals on the ACAP implementors mailing list is strongly encouraged. Registrations MUST provide the following information when requesting a new standard option name.

To: iana@iana.org

Subject: Registration of ACAP standard option

Containing option class: <option-class-name>

Requested option name: <option-class-prefix>.<option-name>

Option short description:

Value semantics: # any specific semantic restrictions placed
on the value of the option, e.g. URL,
integer, ...

Value default: # a recommended default value for the option

[6.](#) Recommendations for standardization

There are no absolute rules for when to register a set of standard options vs. defining a new dataset class. In general, one can always register a standard option class and a set of option names under that class. In some cases, it MAY be appropriate to define a new dataset class instead of registering a set of standard options.

The key distinguishing feature of a dataset class definition is its documentation requirements. Dataset class definitions specify the semantic relationships in the dataset class. The semantic relationships between subdatasets, entries and attributes are fully defined and part of the standard.

If a set of standard options has any of the following characteristics, then it should be considered a candidate for a new dataset class.

1. The option set consists of a list (array) of identically structured options. Proper representation of option sets of this type require a subdataset. This is not prohibited by the standard option namespace, but the list relationship of the options needs to be documented. This makes it appropriate for a new dataset class.
2. Many or all of the options are semantically related or dependent on one another. Individual options are incomplete, or not useful unless evaluated in the context of the other options in the class.

The relationship or dependencies cannot be documented by individual option registrations.

3. The set of options are interpreted by applications with additional semantic rules that apply to the set of options, and cannot be documented in the registration information for individual standard options.

In summary, any standard option set that has semantic relationships beyond a single option value is a candidate for a new dataset class definition. Standard option sets SHOULD NOT be defined that have additional semantic rules outside those of the individual options. Standard option sets that acquire additional semantics over a period of time, SHOULD be redefined as a dataset class using the procedures and format outlined in [[ACAP](#)].

Note that these considerations apply only to options in the "standard" option namespace. Semantic relationships in the vendor namespace are at the discretion of the vendor and are not considered part of the extended standardization content of the ACAP option namespace.

7. Security Considerations

It is important to make sure that access controls are set correctly on personal options. Sensitive configuration information, including application security information, should not be exposed to other users without an explicit request by the user.

This specification does not address storing private certificates and other security information in the option namespace. This may

be added to a future version of this specification when more experimentation has been done.

8. Acknowledgments

Many thanks to the following people who have contributed to development of the option dataset class specification: Jack DeWinter, Rob Earhart, Ned Freed, Randy Gellens, John Meyers, Chris Newman, Lyndon Nerenberg, John-Paul Sicotte, Matt Wall and other participants of the IETF ACAP working group.

Hole, Melnikov

[Page 11]

Internet Draft

ACAP Options

December 2002

9. Normative references

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

<<ftp://ftp.isi.edu/in-notes/rfc2234.txt>>

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

<<ftp://ftp.isi.edu/in-notes/rfc2244.txt>>

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

<<ftp://ftp.isi.edu/in-notes/rfc2119.txt>>

10. Authors' Addresses

Steve Hole
mailto:Steve.Hole@messagingdirect.com

ACI WorldWide/MessagingDirect
900 10117 - Jasper Ave.

Edmonton, Alberta, T5J 1W8, CANADA

Alexey Melnikov

mailto:Alexey.Melnikov@messagingdirect.com

ACI WorldWide/MessagingDirect

59 Clarendon Road, Watford, Hertfordshire,

United Kingdom, WD17 1FQ

11. 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

Hole, Melnikov

[Page 12]

Internet Draft

ACAP Options

December 2002

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.

