Novell Developer Kit

3.9.1

February 28, 2008

www.novell.com

INTEGRATING THIRD-PARTY URL BLOCKING TOOLS WITH NOVELL BORDERMANAGER®

Novell®

Legal Notices

Novell, Inc. makes no representations or warranties with respect to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

Further, Novell, Inc. makes no representations or warranties with respect to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc. reserves the right to make changes to any and all parts of Novell software, at any time, without any obligation to notify any person or entity of such changes.

Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classification to export, re-export or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in the U.S. export laws. You agree to not use deliverables for prohibited nuclear, missile, or chemical biological weaponry end uses. Please refer to Novell Trade Services Policy page (http://www.novell.com/info/exports/) for more information on exporting Novell software. Novell assumes no responsibility for your failure to obtain any necessary export approvals.

Novell, Inc. has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed at Novell Web site legal page (http://www.novell.com/company/legal/patents/) and one or more additional patents or pending patent applications in the U.S. and in other countries.

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

Online Documentation: To access the online documentation for this and other Novell products, and to get updates, see Novell Documentation Web site (http://www.novell.com/documentation).

Novell Trademarks

For Novell trademarks, see the Novell Trademark and Service Mark list (http://www.novell.com/company/legal/trademarks/tmlist.html).

Third-Party Materials

All third-party trademarks are the property of their respective owners.

Contents

About This Guide

This document describes the interface that is used for integrating third-party content filtering applications (such as SurfControl*, LinkWall Suite* or N2H2*) with Novell® BorderManager®. It has the following information.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comments feature at the bottom of each page of the online documentation, or go to Novell Documentation Feedback page (http://www.novell.com/documentation/feedback.html) and enter your comments there.

Documentation Conventions

In Novell documentation, a greater-than symbol (>) is used to separate actions within a step and items in a cross-reference path.

A trademark symbol (®, ™, etc.) denotes a Novell trademark. An asterisk (*) denotes a third-party trademark.

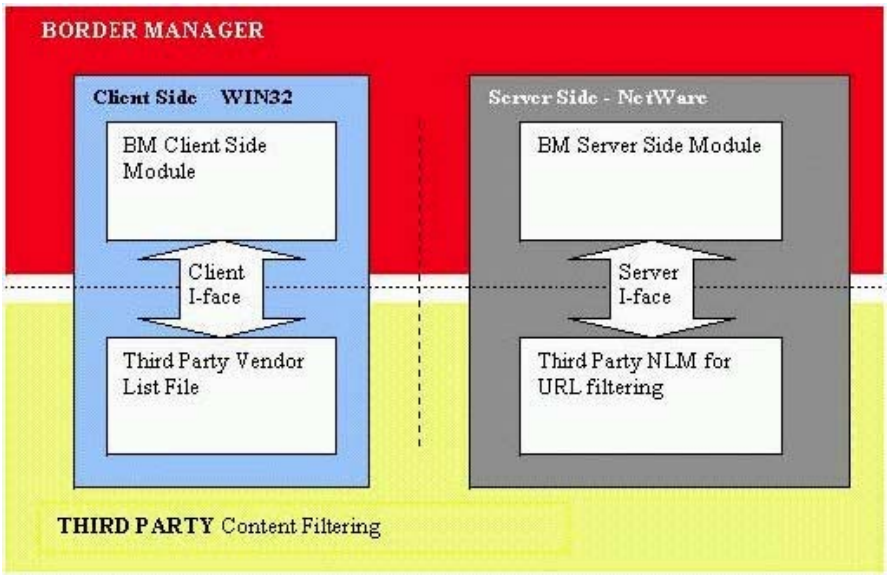
When a single pathname can be written with a backslash for some platforms or a forward slash for other platforms, the pathname is presented with a backslash. Users of platforms that require a forward slash, such as Linux or UNIX, should use forward slashes as required by your software.

Concepts

1

The following illustrates the architecture for integrating third party content filtering with Border Manager®:

Figure 1-1 *Architecture for Integrating Third-Party Content Filtering into BorderManager*



This section has the following information:

- Section 1.1, “Components Involved,” on page 9
- Section 1.2, “What’s New,” on page 10
- Section 1.3, “Interface Specification,” on page 10
- Section 1.4, “Content Filtering Values,” on page 12

1.1 Components Involved

Third party providers supply a configuration file for the client and an NLM™ agent for the server.

- Section 1.1.1, “Client Side: Vendor Registration File,” on page 9
- Section 1.1.2, “Server Side: Vendor’s Agent NLM,” on page 10

1.1.1 Client Side: Vendor Registration File

As depicted in Figure 1-1 on page 9, the client side implementation is in the form of a vendor registration file.

- This file is used to register all URL lists (for example, CyberYES* and CyberNOT* lists) and the list of categories under each list, respectively, to the access control component of the BorderManager client-side snap-in.

- When launched, the access control configuration component reads this registration file and creates a GUI for presenting to the Administrator configuring access control for any source object. (A source object can be any Novell® eDirectory™ object, DNS host name, Host IP address, or Subnet associated with the source of the request for which access needs to be controlled.)
- The configuration file ensures that the third-party application does not require any plug-in for the configuration client.

1.1.2 Server Side: Vendor's Agent NLM

The vendor supplies an NLM that runs on the NetWare® server. The NLM has the following properties:

- This NLM is written by the third party, providing content filtering support to BorderManager access control.
- The main functionality of the NLM is to register a callback function with the BorderManager access control component on the server when coming up.
- When a user tries to access a URL through the proxy, the access control component calls the registered callback function of the agent NLM with the category information and URL.
- The callback function checks its database (implementation dependent) and returns an Allow or Deny.
- When the NLM is unloaded, it must de-register from the access control component.

1.2 What's New

The following changes were made to the API during the Novell BorderManager 3.9 release:

- Access control component allows a 64-bit mask to be stored for each source object, so a maximum of 128 categories is supported.
- Added the registerACLWithProxy128 function to register an agent with the access control component for 128 categories support.
- Added the agentCallBack128 callback function for the proxy server to call, if the agent is registered using the registerACLWithProxy128 API.

1.3 Interface Specification

This section contains the following information:

- Section 1.3.1, "Client Side," on page 10
- Section 1.3.2, "Server Side," on page 12

1.3.1 Client Side

A text file ending with `.acl` extension in `\\server_name\sys\etc\border\lang\*.acl` must be written for registering with the client as a third-party vendor, where lang is the language of the Windows* client.

The FileVersion is specified in the first section. You can specify 1.00 or 2.00 as the file version value. The vendor name, ID, and the product version are described in the next section. ID represents the ID of the third party ACL provider being used. Some of the third-party ID numbers are:

- **SurfControl:** 3
- **N2H2:** 4
- **Connectotel:** 5, 6, 7

Aclcheck.nlm uses the ID to differentiate between the third-party ACL providers being used.

The [Lists] section describes the different kinds of lists available with the product. The example supports two lists, MyNOT and MyYES. The individual categories under MyNOT are described in the [MyNOT] section.

The limit on the number of categories is independent of the number of lists.

After configuration, each source object has an associated mask (OR of the masks of all the categories applicable to that object) stored in eDirectory.

Sample Text File for Version 1.00

If the file version is 1.00, the value must be a power of 2. Thus 1 identifies the sexsites category, 2 identifies sexvideos, 4 identifies techinfo, and 8 identifies techinfomine.

An example text file for version 1.00 is as follows:

```
[File]
FileVersion=1.00

[Vendor]
VendorName=Novell
Version=1.00
ID=3

[Lists]
List1=MyNOT
List2=MyYES

[MyNOT]
sexsites=1
sexvideos=2

[MyYES]
techinfo=4
techinfomine=8
```

The mask of a category is same as the value associated to it, if the FileVersion is 1.00.

Sample Text File for Version 2.00

If the FileVersion is 2.00, the value must be from 0 to 63. The BorderManager 3.9 access control component allows a 64-bit mask to be stored for each source object, so a maximum of 128 categories is supported. The migration is supported only for 64 categories .acl files.

An example text file for version 2.0 is as follows:

```
[File]
FileVersion=2.00

[Vendor]
VendorName=Novell
Version=1.00
ID=3

[Lists]
List1=MyNOT
List2=MyYES

[MyNOT]
sexsites=0
sexvideos=1

[MyYES]
techinfo=2
techinfomine=3
```

If the FileVersion is 2.00, mask associated with it is two to the power of its value.

1.3.2 Server Side

The server side implementation is an agent NLM that uses the API. See Chapter 2, “Functions,” on page 13 and Chapter 3, “Structures,” on page 19.

1.4 Content Filtering Values

Constant	Value
DENY_URL	0
ALLOW_URL	1
ENTRY_NOT_FOUND	2

Functions

2

This documentation alphabetically lists functions and describes their purpose, syntax, parameters, and return values.

deregisterACLFromProxy

Deregisters an agent.

Local Servers: nonblocking

Syntax

```
int deregisterACLFromProxy(
   aclAgent_t   *acl_pt);
```

Parameters

acl_pt

(IN) Points to the aclAgent_t structure that was used during registration.

Return Values

Returns 0 on success.

Remarks

This function is called by agents for deregistration (that is, when they are unloaded).

registerACLWithProxy

Registers an agent with the access control component.

Local Servers: nonblocking

Syntax

```
int registerACLWithProxy(
   aclAgent_t   *acl_pt);
```

Parameters

acl_pt

(IN) Points to the aclAgent_t structure describing the agent's profile and the callback function.

Remarks

When loading, the agent NLM™ calls this function for registering with the access control component. The memory for the aclAgent_t structure must be allocated in the agent NLM.

registerACLWithProxy64

Registers an agent with the access control component for 64 categories support.

Local Servers: nonblocking

Syntax

```
int registerACLWithProxy64(
   aclAgent_t   *acl_pt);
```

Parameters

acl_pt

(IN) Points to the aclAgent_t structure describing the agent's profile and the callback function.

Remarks

When loading, the agent NLM calls this function for registering with the access control component. The memory for the aclAgent_t structure must be allocated in the agent NLM.

registerACLWithProxy128

Registers an agent with the access control component for 128 categories support.

Local Servers: nonblocking

Syntax

```
int registerACLWithProxy128(
   aclAgent_t   *acl_pt);
```

Parameters

acl_pt

(IN) Points to the aclAgent_t structure describing the agent's profile and the callback function.

Remarks

When loading, the agent NLM calls this function for registering with the access control component. The memory for the aclAgent_t structure must be allocated in the agent NLM.

Structures

3

This documentation alphabetically lists structures and describes their purpose and fields.

aclAgent_t

This is the data structure passed when registering with BorderManager® access control.

Syntax

```
typedef struct aclAgent
{
   struct aclAgent  *next;
   struct aclAgent  *prev;
   int   agentID;
   char   agentName[64];
   int (*agentCallBack)(
            int     mask,
            char  *urlName,
            int     urlLen);
   int (*agentCallBack64)(
            unsigned __int64 mask,
            char *urlName,
            int urlLen,
            char *usrName,
            unsigned long ipAddr);
int   (*agentCallBack128)
(unsigned __int64 mask[],
char *urlName,
int urlLen,
char *usrName,
unsigned long ipAddr);
} aclAgent_t;
```

Fields

next

Points to the next agent in the list. This information is used by the proxy cache server.

prev

Points to the previous agent in the list.

agentID

Contains an ID number assigned to the agent by the proxy cache server.

agentName

Contains the name of the provider. This field is filled by the provider.

agentCallBack

Points to the callback function for the proxy server to call if the agent is registered using the registerACLWithProxy API. The provider provides the callback function. For details about this function, see “Remarks” on page 21.

agentCallBack64

Points to the callback function for the proxy server to call, if the agent is registered using the registerACLWithProxy64 API. The provider provides the callback function. For details about this function, see “Syntax” on page 20.

agentCallBack128

Points to the callback function for the proxy server to call, if the agent is registered using the registerACLWithProxy128 API. The provider provides the callback function. For details about this function, see “Remarks” on page 21.

Remarks

The agentCallBack, agentCallBack64 and agentCallBack128 members of the aclAgent_t structure are implemented by the agent NLM™ and registered as the callback functions by calling registerACLWithProxy (page 15), registerACLWithProxy64 (page 16), and registerACLWithProxy128 (page 17) respectively. The callback functions are called by the ACLCHECK.NLM with the appropriate parameters that query the agent NLM regarding ALLOW/DENY access to the URL.

The following parameters are passed to the callback function:

mask

(IN) In case of agentCallBack function it specifies four bytes and in case of agentCallBack64 function it specifies eight bytes to identify the categories applicable to the source object. In case of agentCallBack128, mask[0] contains mask for categories from 0- 63, with LSB (right end) representing the 0th category and MSB (left end) representing the 63rd category. Similarly, mask[1] contains mask from 64 to127 categories, with LSB (right end) representing the 64th category and MSB (left end) representing the 127th category.

urlName

(IN) Points to the URL being accessed. The URL is expressed as a byte array.

urlLen

(IN) Specifies the length of the URL byte array.

usrName

Specifies the user name in the null terminated string format. This parameter is applicable to agentCallBack64 and agentCallBack128 function.

ipAddr

Specifies the four byte client IP address from where the request originated. This parameter is applicable to the agentCallBack64 and agentCallBack128 functions.

The callback returns one of the following:

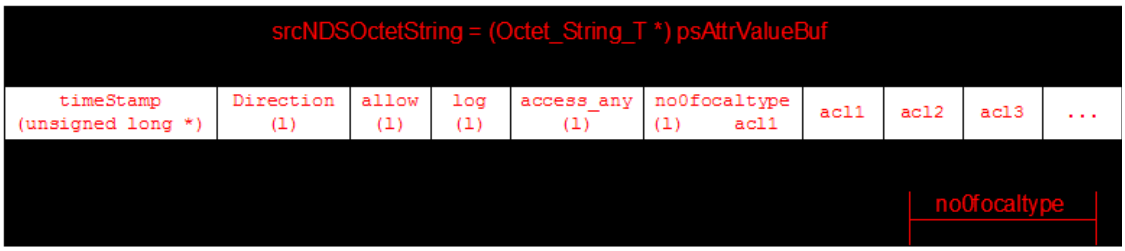
0	DENY_URL
1	ALLOW_URL
2	ENTRY_NOT_FOUND

ACL Rule Format

A

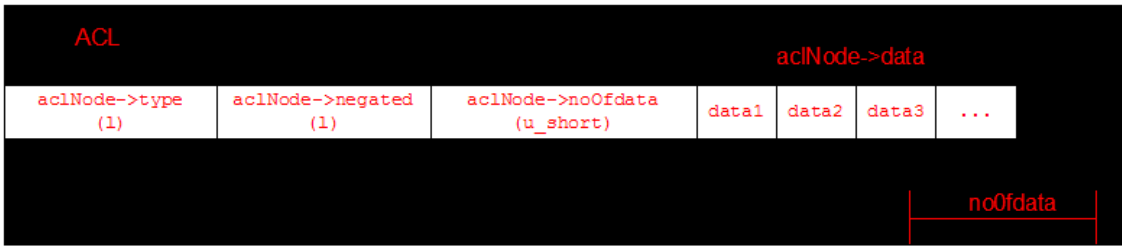
The ACL rules are stored in Novell® eDirectory™ as an Octet String. The attribute name is BRDSRVS:outgoing. Format of the Octet String is shown below:

Figure A-1 *Attribute Value Buffer*



It shows that the first field in the buffer contains the timestamp and the size is an unsigned long. The second field is direction and the size is 1. The rest should be easy to follow. The number of acl fields is determined by the value of noOfacltype. Each acl field contains some information about the act, including the data (see the following figure).

Figure A-2 *ACL Field Diagram*

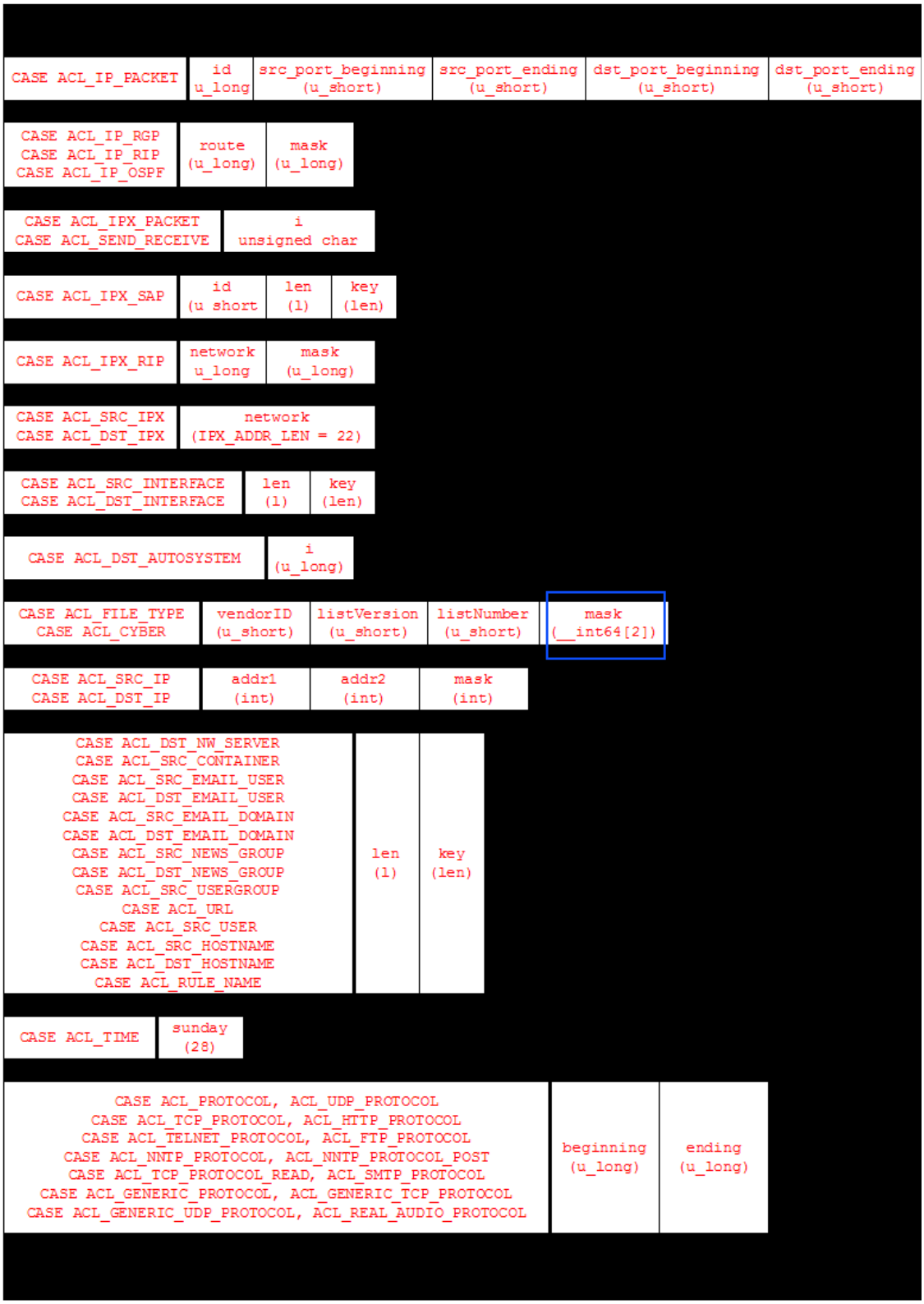


The ACL field contains four major fields:

- type, with a size of 1
- negated
- noOfdata
- data, containing one or more data fields. The number of data fields is determined by the value of noOfdata.

Each data field contains one or more blocks of information about the data itself, depending on the type field. (See the following figure.)

Figure A-3 *acl->data Block Diagram*



The highlighted field is added to the acl data block during the Novell BorderManager 3.9 release to support 128 categories.

If the aclNode->type == ACL_IP_PACKET, the next block in the buffer is the acl id with a size of unsigned long, followed by src_port_beginning with a size of unsigned short, and the rest is typical.

Vendor URL Categorization

B

The ACL Elements are stored in Novell® eDirectory™ as an XML schema. The attribute name is BRDSRVS: ACL Config CurrentXML.

The ACL Schema contains the Access Rules. There can be any number of access rules set for filtering. Each rule corresponds to only one thrid party Vendor. If no vendor is selected then Novell Border Manager uses the native filtering mechanism.

In this Schema, ACL is the root element and AccessRule is a child element of ACL. The AccessRule element has its own child elements. URL is a child element of AccessRule. This element holds URLs to be filtered by BorderManager. URLCategorization is a child element of URL. BorderManager considers this element for categorizations set by the Vendors. Each Vendor has set his own rules on URL categorization.

The following is a schematic representation of ACL > AccessRule > URL and ACL > URLCategorization.

B.1 Access Rule URL elements

Figure B-1 *ACL > AccessRule > URL parent child relationship diagram*

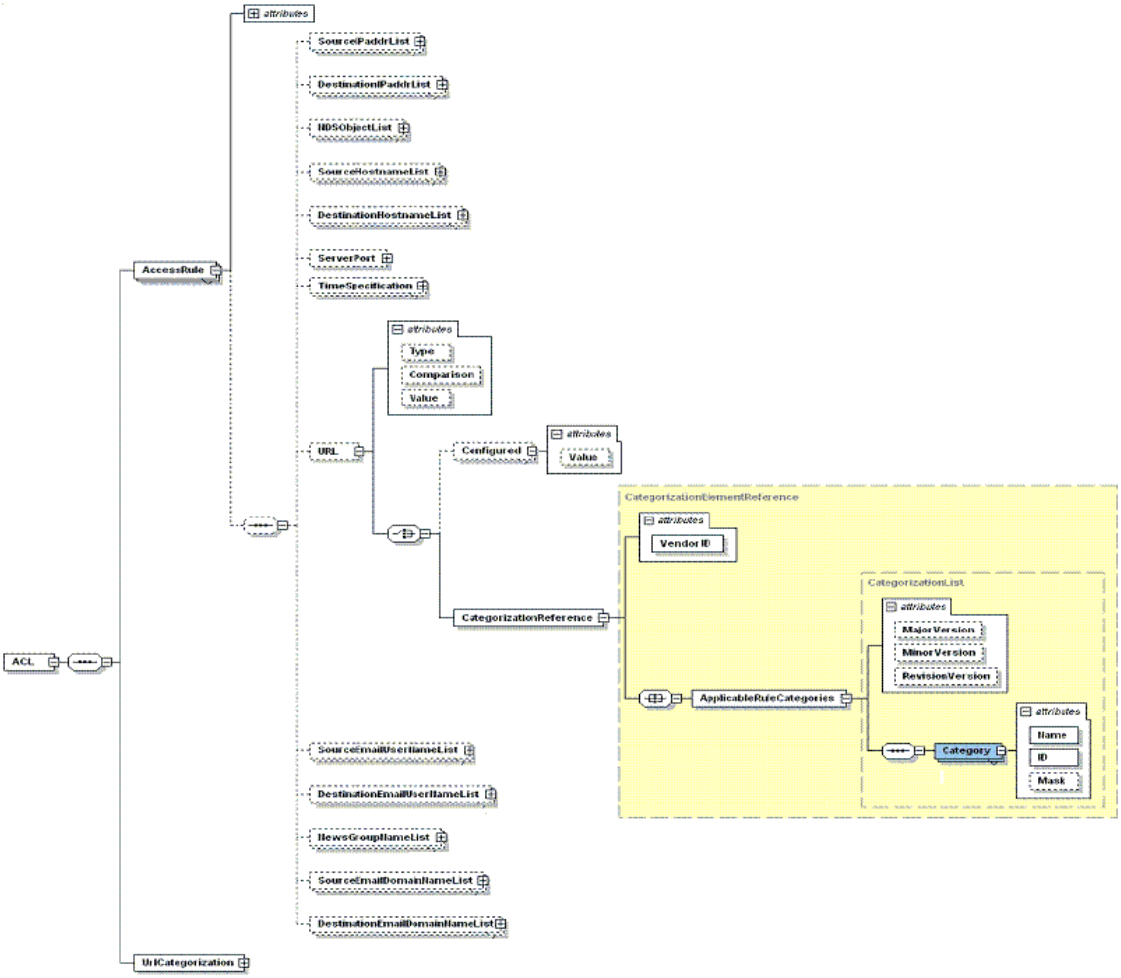


Table B-1 *URL element attribute chart*

Element	Attribute	Data Type	Use	Description
URL			optional	URL is used when an HTTP and FTP services are selected.
Configured	Value	string	optional	This is used internally by the Novell BorderManager.
CategorizationReference	VendorID	string	required	This element tells which rule is matched with which vendor. This attribute holds the unique Vendor ID.

Element	Attribute	Data Type	Use	Description
ApplicableRuleCategories	MajorVersion	integer	required	This holds third party vendor's Major Version.
	MinorVersion	integer	optional	This attribute holds third party vendor's Minor Version.
	RevisionVersion	integer	optional	This attribute holds third party vendor's Revised Version.
Category	Name	string	required	This holds the category name.
	ID	integer	required	This holds the category ID.
	Mask	integer	required	This holds the mask value set by the third party vendor.

B.2 Vendor URL Categorization elements

Figure B-2 *ACL > URLCategorization parent child relationship diagram*

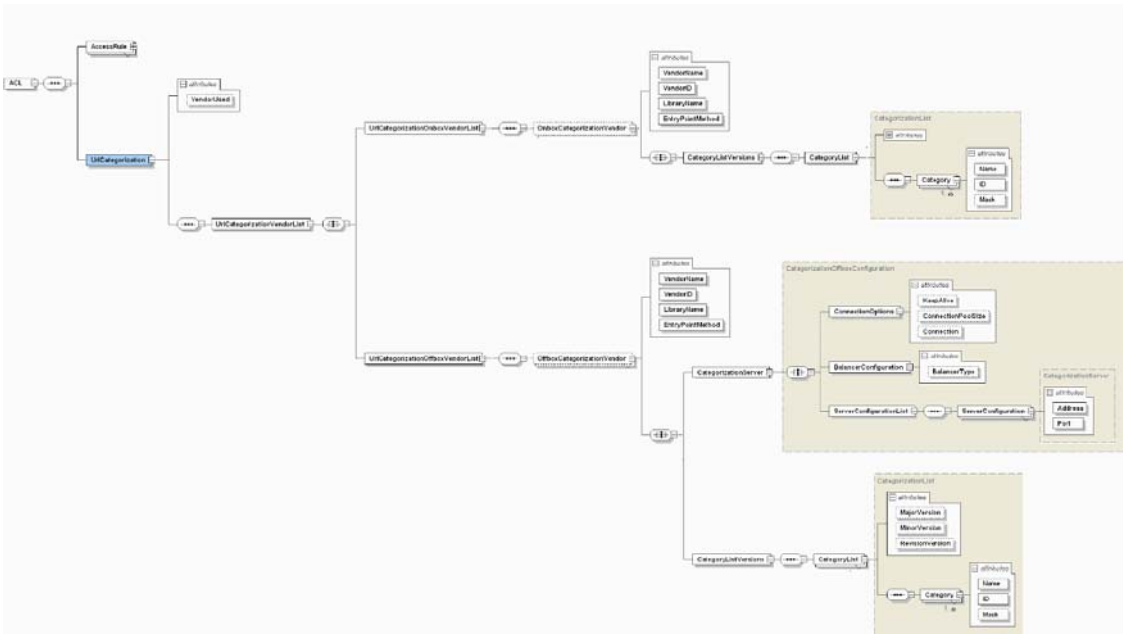


Table B-2 *URL Categorization element attribute chart*

Element	Attribute	Data Type	Use	Description
URLCategorization	VendorUsed	string		This holds the active vendor.

Element	Attribute	Data Type	Use	Description
URLCategorizationVendorList		string		This holds the URL Categorization vendors that Novell BorderManager supports, for example SurfControl, Websense, SecureComputing, and Connectotel.
URLCategorizationOnboxVendor		string		This holds vendors who use Novell Onbox interface to communicate.
OnboxCategorizationVendor	VendorName	string		This holds the third party vendor name.
	VendorID	integer		This holds the thrid party vendor ID.
URLCategorizationOffboxVendor				This holds vendors who use Novell Offbox interface to communicate.
OffboxCategorizationVendor	VendorName	string		This holds the third party vendor name.
	VendorID	integer		This holds the thrid party vendor ID.
CategorizationServer				This holds the list of third party server addresses.
ServerConfigurationList				This holds the Server Configuration List.
ServerConfiguration	Address	string		This holds the Server IP address.
	Port	string		This holds the port number in which Server listens.
CategoryListVersion				This holds the category list of third party vendors. **NOTE:** However, proxy will choose to use only one of them.
CategoryList	MajorVersion	integer	required	There could be one or more category lists supported by the vendor for a major version.

Element	Attribute	Data Type	Use	Description
	MinorVersion	integer	optional	There could be one or more category lists supported by the vendor for a minor version.
	RevisionVersion	integer	optional	There could be one or more category lists supported by the vendor for a revision version.
Category	Name	string	required	This holds the category name.
	ID	integer	required	This holds the category ID.
	Mask	integer	required	This holds the mask value set by the third party vendor.

NOTE: The elements CategoryListVersion, CategoryList, and Category are same for both Onbox and Offbox vendor configurations.

B.3 A Sample URL Categorization Schema

Figure B-3 *A sample URL Categorization schema*

```
- <ACL>
  - <UrlCategorization VendorUsed="SurfControl">
    - <UrlCategorizationVendorList>
      - <UrlCategorizationOnboxVendorList>
        - <OnboxCategorizationVendor VendorID="3" VendorName="SurfControl">
          - <CategoryListVersions>
            - <CategoryList MajorVersion="1.00" MinorVersion="1">
                <Category ID="0" Mask="0" Name="Gambling"/>
                <Category ID="1" Mask="1" Name="Criminal Skills"/>
                <Category ID="2" Mask="2" Name="Hate Speech"/>
                <Category ID="3" Mask="3" Name="Violence"/>
                <Category ID="4" Mask="4" Name="Weapons"/>
                <Category ID="5" Mask="5" Name="Drugs, Alcohol & Tobacco"/>
                . . .
                <Category ID="39" Mask="39" Name="Search Engines"/>
              </CategoryList>
            </CategoryListVersions>
          </OnboxCategorizationVendor>
        - <OnboxCategorizationVendor VendorID="5" VendorName="Connectotel:AdWallList">
          - <CategoryListVersions>
            - <CategoryList MajorVersion="1.2.1" MinorVersion="1">
                <Category ID="0" Mask="0" Name="Finance & Investment"/>
                <Category ID="1" Mask="1" Name="Shopping"/>
                <Category ID="2" Mask="2" Name="Remote Proxies"/>
                <Category ID="3" Mask="3" Name="Games"/>
                . . .
                <Category ID="30" Mask="30" Name="Turn AdWall On"/>
              </CategoryList>
            </CategoryListVersions>
          </OnboxCategorizationVendor>
        - <OnboxCategorizationVendor VendorID="6" VendorName="Connectotel:FileWallList">
          - <CategoryListVersions>
            - <CategoryList MajorVersion="1.00" MinorVersion="1">
                <Category ID="0" Mask="0" Name="Motor Vehicles"/>
                <Category ID="1" Mask="1" Name="Sex Education"/>
                <Category ID="2" Mask="2" Name="Religion"/>
                <Category ID="3" Mask="3" Name="Streaming Media"/>
                . . .
                <Category ID="30" Mask="30" Name="Turn AdWall On"/>
                <Category ID="31" Mask="31" Name="Turn FileWall On"/>
              </CategoryList>
            </CategoryListVersions>
          </OnboxCategorizationVendor>
        - <OnboxCategorizationVendor VendorID="7" VendorName="Connectotel:LinkWallList">
          - <CategoryListVersions>
            - <CategoryList MajorVersion="1.2.1" MinorVersion="1">
                <Category ID="30" Mask="30" Name="Turn LinkWall On"/>
              </CategoryList>
            </CategoryListVersions>
          </OnboxCategorizationVendor>
        </UrlCategorizationOnboxVendorList>
      - <UrlCategorizationOffboxVendorList>
        - <OffboxCategorizationVendor VendorID="4" VendorName="N2H2">
          - <CategorizationServer>
            - <ServerConfigurationList>
                <ServerConfiguration Address="192.65.1.255" Port="4004"/>
              </ServerConfigurationList>
            </CategorizationServer>
          - <CategoryListVersions>
            - <CategoryList MajorVersion="4.00" MinorVersion="1">
                <Category ID="0" Mask="0" Name="Art/Culture/Heritage"/>
                <Category ID="1" Mask="1" Name="Alcohol"/>
                <Category ID="2" Mask="2" Name="Anonymizers"/>
                <Category ID="3" Mask="3" Name="Adult Topics"/>
                <Category ID="4" Mask="4" Name="Anonymizing Utilities"/>
                <Category ID="5" Mask="5" Name="Business"/>
                <Category ID="6" Mask="6" Name="Chat"/>
                . . .
                <Category ID="79" Mask="79" Name="UserDefinedCategory0"/>
                <Category ID="80" Mask="80" Name="UserDefinedCategory1"/>
                <Category ID="81" Mask="81" Name="UserDefinedCategory2"/>
                <Category ID="82" Mask="82" Name="UserDefinedCategory3"/>
                <Category ID="83" Mask="83" Name="UserDefinedCategory4"/>
                <Category ID="84" Mask="84" Name="UserDefinedCategory5"/>
              </CategoryList>
            </CategoryListVersions>
          </OffboxCategorizationVendor>
        </UrlCategorizationOffboxVendorList>
      </UrlCategorizationVendorList>
    </UrlCategorization>
  - <AccessRule Action="1" RuleNumber="473b396" Log="0" RuleLocation="string" Current_RuleName="Rule:473b396" RuleName=""
    License="0" Direction="0" AccessType="HTTP">
    - <URL Type="SurfControl" Value="None" Comparison="Onbox">
      - <CategorizationReference VendorID="3">
        - <ApplicableRuleCategories MajorVersion="1.00" MinorVersion="1">
            <Category Name="Travel" ID="8" Mask="8"/>
```

Revision History

C

The following table lists changes made to this documentation in reverse chronological order:

Release	Changes
February, 2008	Added a new function registerACLWithProxy128 (page 17).
	Added a new field “agentCallBack128” on page 21.
	Added a new Appendix Appendix B, “Vendor URL Categorization,” on page 27
October, 2006	Updated Section 1.3.1, “Client Side,” on page 10.
	Added a new function registerACLWithProxy64 (page 16).
	Added a new field “agentCallBack64” on page 21.
July, 2003	Added to the NDK as a Leading Edge component.