
HANDLE.NET (version 6.2)

Technical Manual

(Version 2)

**Corporation for National
Research Initiatives
March 2007
hdl:4263537/5031**

HANDLE.NET 6.2 software is subject to the terms of the Handle System Public License (version 2). Please read the license: <http://hdl.handle.net/4263537/5030>.

A Handle System Service Agreement is required in order to provide identifier/resolution services using the Handle System technology. Please read the Service Agreement: <http://hdl.handle.net/4263537/5029>.

© Corporation for National Research Initiatives, 2006, All Rights Reserved.

Revision History

Version	Publication Date	Summary of Changes and Updates
1	06/16/2006	Initial Draft
2	03/30/2007	SimpleSetup Tool example output showing configuration options added. Instructions for using PostgreSQL as custom handle storage added. Information on handle type values clarified and updated.

Credits

CNRI wishes to thank the prototyping team of the Los Alamos National Laboratory Research Library for their collaboration in the deployment and testing of a very large Handle System implementation, leading to new designs for high performance handle administration, and handle users at the [Max Planck Institute for Psycholinguistics](#) and [Lund University Libraries NetLab](#) for their instructions for using PostgreSQL as custom handle storage.

Table of Contents

1	Introduction	6
1.1	Handle Syntax.....	6
1.2	Handle System Scalability.....	7
1.2.1	Storage.....	7
1.2.2	Performance.....	7
1.3	Authentication	8
1.3.1	Types of Authentication.....	8
1.3.2	Certification	9
1.3.3	Sessions.....	9
1.3.4	Algorithms	9
2	Installing and Running a Handle Server.....	10
2.1	Installing Java™	10
2.2	Unpacking the Distribution	10
2.3	Choosing an Installation Directory	10
2.4	Running the Setup Program.....	11
2.4.1	SimpleSetup Tool.....	11
2.5	Running the Handle Server for the First Time.....	14
2.6	How Your Prefix is Set Up.....	15
2.7	Installation Directory.....	15
2.7.1	access.log	15
2.7.2	error.log.....	16
2.7.3	config.dct.....	16
2.7.4	\$HOME/root info	17
2.7.5	cache.jdb.....	17
2.7.6	handles.jdb.....	17
2.7.7	nas.jdb.....	17
2.7.8	dbtxns.log.....	17
2.7.9	siteinfo.bin	17
2.7.10	admpub.bin, admpriv.bin	17
2.7.11	pubkey.bin, privkey.bin	17
2.7.12	txns	17
2.8	Splitting a Handle Server	18
2.9	Shutting Down a Handle Server	20
2.10	Inactive Prefixes.....	20
3	Handle Administration Tools	21

3.1	The Admin Tool.....	21
3.1.1	Query Handle	22
3.1.2	Create Handle	23
3.1.3	Modify Handle	27
3.1.4	Remove Handle	28
3.1.5	Running Batch Files	29
3.1.6	"Home" a Prefix.....	31
3.1.7	"Unhome" a Prefix.....	31
3.1.8	Backing Up a Handle Server	32
3.1.9	Listing Handles on a HANDLE.NET Server	33
3.1.10	Authentication	33
3.1.11	Generate Key Pairs	33
3.1.12	Using Sessions	34
3.1.13	Console.....	35
3.2	The Handle Tool.....	36
3.2.1	Installation	36
3.2.2	Default Window	36
3.2.3	Console	38
3.2.4	Authentication	39
3.2.5	Lookup	41
3.2.6	Create, Modify or Delete Handles	43
3.2.7	Modify.....	48
3.2.8	Remove Handle Value	50
3.2.9	Delete Handle	50
3.2.10	Batch Processing.....	50
3.2.11	Handle Value Line Format	53
3.2.12	Using the Batch Processor Tool	55
3.2.13	Homing/Unhoming a Prefix.....	58
4	Batch Operation - Command Line.....	60
4.1	Create Handle Batch Format	60
4.2	Delete Handle Batch Format.....	61
4.3	(Un)Home Prefix Batch Format.....	61
4.4	Add Handle Value Batch Format.....	61
4.5	Remove Handle Value Batch Format.....	62
4.6	Modify Handle Batch Format.....	62
4.7	Authentication Information Format	62
4.8	Session Setup Information Format.....	63

4.9	Handle Value Line Format	64
5	Advanced Server Configuration	67
5.1	hdl_http_config.....	67
5.2	hdl_tcp_config.....	68
5.3	hdl_udp_config.....	68
5.4	server_config.....	68
5.5	log_save_config	70
5.6	Other Settings	70
5.7	Example config.dct File.....	71
6	Other Tools and Features	73
6.1	DBTool	73
6.2	DBList/DBRemove	73
6.3	Query Resolver	73
6.4	Test Tool	73
6.5	KeyUtil.....	75
6.6	GetRootInfo.....	75
6.7	GetSiteInfo	75
6.8	DoCheckpoint.....	75
7	Replication	76
7.1	Setting up a Single Mirror Handle Server.....	76
7.2	Setting up a Second Mirror Handle Server	76
8	Using Custom Handle Storage.....	77
8.1	Enabling Berkeley DB JE Database Support.....	77
8.2	Using a SQL Database for Storage.....	78
8.2.1	Configuring the Handle Server	78
8.2.2	Example SQL Tables	79
8.2.3	Depositing Handles Outside the Handle Server.....	80
8.2.4	Using Custom SQL Statements	80
8.3	Using PostgreSQL Database	85
9	The Handle HTTP Proxy	87
9.1	Using the Proxy.....	87
9.2	Using Custom HTML Pages.....	89
9.3	Web Interface for Handle Administration	89
10	Handle Clients & the Client Library (Java™ Version)	92
11	Configuring an Independent Handle Service	94
11.1	Client Configuration Details.....	94
11.2	Server Side Configuration	95

1 Introduction

The Handle System is a comprehensive system for assigning, managing, and resolving persistent identifiers for digital objects and other resources on the Internet. The Handle System includes an open set of protocols, a namespace, and an implementation of the protocols. The protocols enable a distributed computer system to store identifiers of digital resources and resolve those identifiers into the information necessary to locate and access the resources. This associated information can be changed as needed to reflect the current state of the identified resource without changing the identifier, thus allowing the name of the item to persist over changes of location and other state information.

1.1 Handle Syntax

Within the handle namespace, every identifier consists of two parts: its prefix and a unique local name under the prefix, otherwise known as its suffix. The prefix and suffix are separated by the ASCII character "/". A handle may thus be defined as

```
<Handle> ::= <Prefix> "/" <Handle Local Name>
```

For example, handle "12345/hdl1" is defined under the Handle Prefix "12345", and its Handle Local Name is "hdl1".

Handles may consist of any printable characters from the Universal Character Set, two-octet form (UCS-2) of ISO/IEC 10646, which is the exact character set defined by Unicode v2.0. The UCS-2 character set encompasses most characters used in every major language written today. To allow compatibility with most of the existing systems and prevent ambiguity among different encoding, handle protocol mandates UTF-8 to be the only encoding used for handles. The UTF-8 encoding preserves any ASCII encoded names, which allows maximum compatibility to existing systems without causing naming conflict.

By default, handles are case sensitive. However, any handle service, including the Global Handle Registry (GHR), may define its namespace such that all ASCII characters within any identifier are case insensitive.

The handle namespace can be considered as a superset of many local namespaces, with each local namespace having its own unique handle prefix. The prefix identifies the administrative unit of creation, although not necessarily continuing administration, of the associated handles. Each prefix is guaranteed to be globally unique within the Handle System. Any existing local namespace can join the Global Handle namespace by obtaining a unique prefix, with the resulting identifiers being a combination of prefix and local name as shown above.

Each prefix may have many derived prefixes registered underneath. Any derived prefix can only be registered by its parent after its parent prefix is registered. Every handle is then defined under a prefix. The prefix and the local name are separated by the octet used for ASCII character "/" (0x2F). The collection of local names under a prefix is the local namespace for that prefix. Any local name must be unique under its local namespace. The uniqueness of a prefix and a local name under that prefix ensures that any identifier is globally unique within the context of the Handle System.

1.2 Handle System Scalability

Scalability was a critical design criterion for the Handle System. The problem can be divided into storage and performance. That is, is there some limit to the number of identifiers that can be added? And, does performance go down, or do some functions simply break with increased numbers of identifiers, such that at some point the system becomes unusable? Specific details on this are given below, but it is important to keep two higher level issues in mind. First, it is important here, as in many other places, to distinguish between Handle System design and any given implementation. Scalability in design may or may not work out as expected in any given implementation, but if the design is fundamentally scalable, specific implementation problems can be corrected as they are encountered. Secondly, use of the Handle System through some other service, e.g., an http proxy, may well introduce other scalability issues which the basic Handle System design does not and cannot address.

1.2.1 Storage

The Handle System has been designed at a very basic level as a distributed system; that is, it will run across as many computers as are required to provide the desired functionality.

Handles are held in and resolved by handle servers and the handle servers are grouped into one or more handle sites within each handle service. There are no design limits on the total number of handle services which constitute the Handle System, there are no design limits on the number of sites which make up each service, and there are no limits on the number of servers which make up each site. Replication by site, within a handle service, does not require that each site contain the same number of servers; that is, while each site will have the same replicated set of handles, each site may allocate that set of handles across a different number of servers. Thus increased numbers of handles within a site can be accommodated by adding additional servers, either on the same or additional computers, additional sites can be added to a handle service at any time, and additional handle services can be created. Every service must be registered with the Global Handle Registry, but that handle service can also have as many sites with as many servers as needed. The result is that the number of identifiers that can be accommodated in the current Handle System is limited only by the number of computers available.

1.2.2 Performance

Constant performance across increasing numbers of identifiers is addressed by hashing, replication, and caching.

Hashing, a technique well known to database designers, is used in the Handle System to evenly allocate any number of identifiers across any number of servers within a site, and allows a single computation to determine on which server within a set of servers a given handle is located, regardless of the number of handles or the number of servers. Each server within a site is responsible for a subset of handles managed by that site. Given a specific identifier and knowledge of the handle service responsible for that identifier, a handle client selects a site within that handle service and performs a single computation on the handle to determine which server within the site contains the handle. The result of the computation becomes a pointer into a hash table, which is unique to each handle site and can be thought of as a map of the given site, mapping which handles belong to which servers. The computation is independent of the number of servers and handles, and it will

not take a client any longer to locate and query the correct server for a handle within a handle service that contains billions of handles and hundreds of servers, than for a handle service that contains only millions of handles and only a few servers.

The connection between a given handle and the responsible handle service is determined by prefix. Prefix records are maintained by the Global Handle Registry as handles, and these handles are hashed across the Global Handle Registry sites in the same way that all other handles are hashed across their respective handle services. The only hierarchy in Handle System services is the two level distinction between the single Global and all locals, which means that the worst case resolution would be that a client with no built in or cached knowledge would have to consult Global and one local.

Another aspect of Handle System scalability is replication. The individual handle services within the Handle System each consist of one or more handle service sites, where each site replicates the complete individual handle service, at least for the purposes of handle resolution. Thus, increased demand on a given handle service can be met with additional sites, and increased demand on a given site can be met with additional servers. This also opens up the option, so far not implemented by any existing clients, of optimizing resolution performance by selecting the "best" server from a group of replicated servers.

Caching may also be used to improve performance and reduce the possibility of bottleneck situations in the Handle System, as is the case in many distributed systems. The Handle System data model and protocol design includes a space for cache time-outs and handle caching servers have been developed and are in use. (See RFC 3651, [Handle System Namespace and Service Definition](#), Section 4.2, Handle System Middleware Components, hdl:4263537/4068.)

1.3 Authentication

The current distribution of the HANDLE.NET software uses the [Sun Java™ Cryptography Extension and Provider](#) for low level cryptography routines. This library comes standard with Java™ version 1.4 .

1.3.1 Types of Authentication

As further described in *System Fundamentals: Authentication* on the Handle System web site, the Handle System provides two forms of authentication: public key and secret key.

In the current implementation, public key authentication is performed using the DSA algorithm. The key length is variable from 512 to 1024 bits, and can be chosen by the user when generating keys. The HANDLE.NET software distribution defaults to a 1024 bit key.

Public key authentication requires two keys: a public key and a private key. The public key is stored in a handle as it should be available to the public. The private key should be securely stored on the computer with the handle client that will be authenticated. To prevent unauthorized use of a private key it can be encrypted using a symmetric algorithm. The current implementation of the Handle System uses 56 bit DES for this purpose.

Secret key authentication relies on a secure hashing algorithm, chosen by the client being authenticated. Currently, this algorithm can be either MD5 or SHA1. A secret key consists of a single byte string of size ranging from 0 to 2147483648. This byte string is stored as plain

text in a handle. It is highly advisable to restrict read permissions on the handle to ensure the secrecy of the secret key.

1.3.2 Certification

Clients can request that a handle server cryptographically certify its messages with its public key. This certification can be used to verify the authenticity of handle server transmissions. The current implementation of the Handle System uses DSA for this purpose. The DSA public key for a handle server is stored in its site information record.

1.3.3 Sessions

Establishing sessions with a handle server offers additional security functionality. For background on handle server sessions, see *System Fundamentals: [Sessions](#)* on the Handle System web site.

The Handle System allows for encryption of communication after establishing a session with a handle server. This is equivalent to SSL or TLS as used in protocols such as HTTPS, as it affords protection from eavesdropping and man-in-the-middle attacks. The current implementation of the Handle System encrypts session communications using 56bit DES.

For instructions on enabling session encryption see Chapter 3, *Using the Admin Tool*, and Chapter 4, *Batch Operation*.

1.3.4 Algorithms

More information on the algorithms mentioned above can be found at the locations below.

DSA: <http://www.itl.nist.gov/fipspubs/fip186.htm>

DES: <http://www.itl.nist.gov/fipspubs/fip462.htm>

2 Installing and Running a Handle Server

This section outlines the steps required to get the HANDLE.NET software, install it, set it up, and acquire a prefix.

2.1 Installing Java™

The HANDLE.NET software requires Java™ to run. If you already have Java™ installed, confirm that you have version 1.4.2. You can check this by running the command

```
'java -version'
```

If you need to install Java™, you can obtain a version for several popular platforms at <http://java.sun.com/j2se/>. For other platforms, contact your vendor.

2.2 Unpacking the Distribution

Download the HANDLE.NET software distribution from <http://www.handle.net>. To uncompress the distribution package on Unix-like platforms use the `gunzip` and `tar` programs. The distribution package can be uncompressed on Win32 based systems using the program WinZip®.

The distribution will uncompress to the following files:

```
admintool.jar
apidoc (directory)
handle.jar
hdlnet2lic.txt
hdlnet2sa.txt
history.txt
install.txt
jclientlib6lic.txt
src.zip
```

Copy the jar file ('handle.jar') into a separate directory on your computer. For the rest of these instructions, we will assume this directory is '/hs/bin/'.

2.3 Choosing an Installation Directory

Choose a directory that will hold the configuration and data for this server. For the rest of these instructions we will use the '/hs/svr_1' directory. Be sure to create a new directory for each server on the same machine. To create this directory on Unix, run this command:

```
mkdir -p /hs/svr_1
```

For more information on the files contained in the installation directory, see Chapter 8, *Installation Directory*.

2.4 Running the Setup Program

The HANDLE.NET software includes an installation program. The program requires Java™, so make sure you have the java binary directory in your system path.

On Unix-like systems it can be invoked by running the command:

```
java -cp /hs/bin/handle.jar net.handle.server.SimpleSetup /hs/svr_1
```

On Windows, DOS, or OS/2like systems:

```
java -cp \hs\bin\handle.jar net.handle.server.SimpleSetup \hs\svr_1
```

The SimpleSetup tool asks for the log rotation interval and sets up the config.dct file properly, but omits the log_save_directory and the Weekly rotation option, leaving those as manual settings that can be made by those who wish to edit the config.dct file by hand.

The installation program guides you through a series of configuration options. Once complete, there will be a file called 'sitebndl.zip' in your handle server directory which you will email to the Handle System Administrator at hldadmin@cnri.reston.va.us. The Administrator will use the 'sitebndl.zip' file to create a prefix in the root service (known as the Global Handle Registry), and will notify you when this has been completed. You will not be able to continue the install until you receive further information from the Handle System Administrator concerning your prefix.

Please note: A Handle System Service Agreement is required in order to provide identifier/resolution services. (Please see <http://hdl.handle.net/4263537/5029> for more information.) You will not receive a prefix from the Handle System Administrator until you have agreed to the terms of the Service Agreement. Once you have agreed to the Service Agreement, paid the appropriate fees, and received prefix information from the Handle System Administrator, you may proceed with the following steps to 'Home' your prefix to your new handle service.

2.4.1 SimpleSetup Tool

Below is an example of the output generated during a SimpleSetup session, showing the configuration options. The file generated by running the tool, sitebndl.zip, is used to create a prefix in the Global Handle Registry.

```
To configure your new handle server, please answer the
questions which follow; default answers, shown in [square
brackets] when available, can be chosen by pressing Enter.
```

```
Will this be a regular handle server or caching server?
```

- 1 - Regular Handle Server (recommended)
- 2 - Caching Handle Server

```
Please choose 1 or 2 and press Enter [1]: 1
```

Will this be a "primary" server (i.e., not a mirror of another server)?(y/n) [y]

Through what IP address will this server be accessible? [*<your_IP>*]:

Enter the (TCP/UDP) port number this server will listen to [2641]:

What port number will the HTTP interface be listening to? [8000]:

Would you like to log all accesses to this server?(y/n) [n]:

Please indicate whether log files should be automatically rotated, and if so, how often.

("N" (Never), "M" (Monthly), "W" (Weekly), or "D" (Daily))?
[Never] : M

NOTE: Auto-saves and restarts will be done on the first of each month.

Select a time of day for the saving and restarting (HH:MM:SS) [00:00:00]: 12:00:00

Enter the full pathname of the directory where saved logs should be stored [.] : /serv/logs

Each handle site has a version/serial number assigned to it. This is so that a client can tell if a particular site's configuration has changed since the last time it accessed a server in the site. Every time you modify a site (by changing an IP address, port, or adding a server, etc), you should increment the version/serial number for that site.

Enter the version/serial number of this site [1]:

Please enter a short description of this server/site: *<Description HERE.>*

Please enter the name of your organization: *<ORG NAME>*

Please enter the name of a contact person for ORG NAME (optional) [(none)]: *<CONTACT NAME>*

Please enter the telephone number of CONTACT NAME or of ORG NAME (optional) [(none)]

Please enter the email address of CONTACT NAME or of ORG NAME: *<contact_name@org_name>*

The Handle System can communicate via UDP and/or TCP sockets. Since UDP messages are blocked by many network firewalls, you may want to disable UDP services if you are behind such a firewall.

Would you like to disable UDP services?(y/n) [n]:

Generating keys for: Server Certification

The private key that is about to be generated should be stored in an encrypted form on your computer. Encryption of the private key requires that you choose a secret passphrase that will need to be entered whenever the server is started. Your private key may be stored unencrypted. Please take all precautions to make sure that only authorized users can read your private key.

Would you like to encrypt your private key?(y/n) [y]: y

Please enter the private key passphrase for Server Certification:

Note: Your passphrase will be displayed as it is entered

oneexample

Please re-enter the private key passphrase:

Note: Your passphrase will be displayed as it is entered

oneexample

Generating keys for: Administration

The private key that is about to be generated should be stored in an encrypted form on your computer. Encryption of the private key requires that you choose a secret passphrase that will need to be entered whenever the server is started. Your private key may be stored unencrypted. Please take all precautions to make sure that only authorized users can read your private key.

Would you like to encrypt your private key?(y/n) [y]:

Please enter the private key passphrase for Administration:

Note: Your passphrase will be displayed as it is entered

anotherexample

Please re-enter the private key passphrase:

Note: Your passphrase will be displayed as it is entered

anotherexample

Generating site info record...

You have finished configuring your regular (primary) handle service. This service now needs to be registered in the Global Handle Registry(GHS).

Please go to <http://hdl.handle.net/4236567/5014> to upload

your newly created sitebndl.zip file. Please carefully read the directions on this page. When the handle administrator receives your file, a prefix will be created and you will receive notification via email.

Please send all questions to hdladmin@cnri.reston.va.us. Thank you for your interest in CNRI's Handle System.

2.5 Running the Handle Server for the First Time

Go to your 'svr_1' directory (where you installed your HANDLE.NET software) and edit the 'config.dct' file. Replace the words `YOUR_NAMING_AUTHORITY` under `server_admins` and `replication_admins` with your prefix (as indicated in your email message). This allows anyone with the private key that matches your public key to be an administrator for your server.

Start the server using the configuration just generated. This can be done with the following command:

On Unix-like systems:

```
java -cp /hs/bin/handle.jar net.handle.server.Main /hs/svr_1
```

On Windows, DOS, or OS/2like systems:

```
java -cp \hs\bin\handle.jar net.handle.server.Main \hs\svr_1
```

Note: If you chose to encrypt your private key(s), you will be prompted for your passphrase here. Also note that Java™ does not have the ability to disconnect from a terminal so you will have to put the process in the background. On Unix systems type `Ctrl Z`, then `bg`, then press `ENTER`.

Next, start the Admin Tool using the following command. (See Chapter 3.2 for instructions if you prefer to use the Handle Tool instead.) This assumes your 'handle.jar' file is in the '/hs/bin' directory.

```
java -cp /hs/bin/handle.jar net.handle.apps.gui.hadmin.HandleTool
```

Click on the 'Server Admin' button and choose 'Home Naming Authority' from the menu that pops up. You will be prompted for your authentication information.

The authentication type should be Public Key.

The ID Handle will be 0.NA/YourPrefix.

The Index should be 300.

Browse to find your private key file. It will be in the 'svr_1' directory (where you installed your server) and is named 'admpriv.bin'. You will be prompted for your secret passphrase. This is the password you entered for Administration when you ran the setup program.

After you are successfully authenticated, a window will appear. Type in your handle server address (IP address *). Type in your prefix by replacing YOUR_NAMING _AUTHORITY. If you used the default port then you should not have to change the port number. Your interface will most likely be TCP. Then press 'OK'.

You should receive a success message. You are now ready to create identifiers in your local HANDLE.NET server.

****Please note that the Handle System does not use DNS.***

2.6 How Your Prefix is Set Up

This section describes how your prefix was set up and how authentication and permissions can be configured.

The prefix 12345 would have the following properties:

```
Prefix Handle: 0.NA/12345
Prefix Admin Group: 0.NA/12345 index 200
Prefix Public Key: 0.NA/12345 index 300
```

When authenticating you will need to identify yourself using the Prefix Admin Public Key and the associated private key which is in your 'admpriv.bin' file on your computer. The instructions for this are in the 'INSTALL' file (see Appendix I) and above in Section 2.5, *Running the HANDLE.NET Server For the First Time*. This is what you used to authenticate yourself to "home" your prefix.

When creating new identifiers, you will need to specify an administrator who will have permission to modify or delete each new identifier (handle). This is done by adding an Admin value that references a public key, secret key or admin group to each new handle. We recommend that you specify your NA Admin Group (see above) value as the administrator for each new handle.

Every value in a handle has a different index. The following pattern works well. Start with 100 for all admin values, start admin group values at 200 and make the public key index 300. So the values of a handle (12345/hdl1), might look like this:

```
100 HS_ADMIN 0.NA/12345 index 200
3 URL http://www.someorg.com/info
4 email someone@someorg.com
```

2.7 Installation Directory

The installation directory contains a number of files. This section explains the function of each.

2.7.1 access.log

If you enabled 'log accesses' on any of your handle server interfaces, all requests sent to those interfaces are logged here. Below is a sample line from this file.

```
10.27.4.211 TCP:HDL "Tue Dec 18 18:48:33 EST 2001" 101 1 111ms 200/8
```

The first column is the IP address of the host that made the request. The second column shows the interface the request was made on. Next is the date and time the request was made. The time is followed by the Handle Operation Requested Code (OP) of the request. In this case the OP is 101, or delete handle. The OP is followed by the Handle Server Response Code (RC). In this case, the RC is 1, or success. Here is a list of possible OPs and RCs:

Handle Operation Requested Code (OP)		Handle Server Response Code (RC)	
1	Query Handle	1	Success
100	Create Handle	2	Error
101	Delete Handle	3	Server Too Busy
102	Add Value	4	Protocol Error
103	Remove Value	5	Operation Not Supported
104	Modify Value	6	Recursion Count Too High
105	List Handles	7	Authentication Error
200	Challenge Response	100	Handle Not Found
201	Verify Response	101	Handle Already Exists
400	Session Setup	102	Invalid Handle
401	Session Terminate	200	Values Not Found
402	Session Exchange Key	201	Value Already Exists
1003	Backup Server	300	Out of Date Site Info
		301	Server Not Responsible
		302	Service Referral
		303	Server Backup
		400	Invalid Admin
		401	Insufficient Permissions
		402	Authentication Needed
		403	Authentication Failed
		404	Invalid Credential
		405	Authentication Timed Out
		406	Authentication Error
		500	Session Timeout
		501	Session Failed
		502	Invalid Session Key
		504	Invalid Session Setup Request

After the rc code, the log entry shows the number of milliseconds the server took to respond to the request. This is useful for gauging the performance of a handle server. The final column of the log entry indicates the handle that was requested (if applicable).

2.7.2 error.log

As suggested by its name, this file contains a log of server errors.

2.7.3 config.dct

This is the server configuration file. See Chapter 5 *Advanced Server Configuration*, for more information.

2.7.4 \$HOME/root info

The 'root_info' file contains the 'HS SITE' records for the global handle servers, necessary for handle resolution. This file isn't actually stored in the handle server directory, but in the subdirectory '.handle' under the home directory of whatever user runs the handle server.

2.7.5 cache.jdb

This is a database used by the server as a temporary handle cache.

2.7.6 handles.jdb

This is the main database for the handle server. It stores all handles and handle values.

2.7.7 nas.jdb

This is the prefix database for the handle server. It stores all prefixes that are homed to the server.

2.7.8 dbtxns.log

This is a log of database transactions, and it is used for disaster recovery. It may grow large in size. It may be deleted at any point to free up disk space, but once it's gone, there will be no way to recover if the handle database becomes corrupted.

See Section 3.8, *Backing Up a Handle Server* for more information.

2.7.9 siteinfo.bin

This contains the 'HS SITE' record for this server. It is stored in the handle prefixes that this server is responsible for.

2.7.10 admpub.bin, admpriv.bin

These are the public and private keys that were created for the administrator during the installation process.

2.7.11 pubkey.bin, privkey.bin

These are the public and private keys for the server. The public key is stored in the servers 'HS SITE' entry. The private key is used to sign responses to requests.

2.7.12 txns

This directory stores the server transaction queue. This keeps track of handle administration operations in order to replicate to secondary servers.

The transaction queue is separated into separate files so that it will be easy to wipe out old transactions. The old files may be deleted in order to free up disk space.

Individual records within each transaction log look something like this:

```
<CRLF><txnID>|<action>|<date>|<hashOnAll>|<hashOnNA>|    <hashOnId>|<handlehex-  
encoded>|
```

2.8 Splitting a Handle Server

This chapter describes how to split a single handle server into multiple handle servers. This does not describe how to split sites consisting of multiple servers. (That is a much more complicated process. Contact hldadmin@cnri.reston.va.us for assistance.)

The goal of this procedure is to minimize the down time for the primary site. The database splitting process can be performed on a checkpoint/backup of the source database, and then completed using the transaction log of the source database.

Here are the steps for splitting a single server site:

1. Create the directories and configuration files for the new servers. Do this by running the `net.handle.server.SimpleSetup` program to create a new directory, config file, and `'siteinfo.bin'` for each new server.
2. Combine the `'siteinfo.bin'` files for each of the new servers into one `HS_SITE` record using the Handle Admin tool:
 - i. Start the Handle Admin tool.
 - ii. Click the 'Create Handle' button to open a create-handle window.
 - iii. Click the 'Add Custom Data' button.
 - iv. Select the `'HS_SITE'` value type, and click the 'Value Data' button.
 - v. Set the protocol for the new site info to 2 and 1.
 - vi. Set the serial # for the new site to something greater than the serial number of the existing server.
 - vii. Enter the information for each new server (IP address, the new server's public key and ports) into the site-info window (this information is in the server directory) by clicking the 'Add' button in the 'Servers' section of the site-info window.
 - viii. Set the `ServerID` for each server to a unique number. The number will be specified in the `'config.dct'` file for each server, so that the server knows which server it is, relative to the other servers in the `'siteinfo.bin'` file.

Save the newly created site-info value to a new `'siteinfo.bin'` file. Copy this new file into the installation directory for each of the new servers. Also make sure to add the `this_server_id` setting to the `server_config` section of each new server's `config.dct` file.

3. Run a checkpoint/backup on the server that is to be split. Then wait until the checkpoint/backup operation is complete. The old primary server should still be running at this point.
4. Decrease the timeout/TTL value for the 'HS_SITE' value that is being modified in the prefix or service handle. A good setting is probably 1200 (20 minutes). Do this so that when you change the 'HS_SITE' value, clients don't still try to access the old server for an entire day after the 'HS_SITE' value is changed.
5. Run the `net.handle.apps.tools.SplitServer` program to split the 'handles.bak' file into the new server directories. Make sure to specify the new server directories in the correct order on the command line. For example:

```
java -cp handle.jar net.handle.apps.tools.SplitServer
/usr/local/current_hs/handles.bak /usr/local/new_hs1
/usr/local/new_hs2 /usr/local/new_hs3
```

This phase can take a long time when splitting large databases. If the database is large enough the process may even take several days. The old server should still be running at this point.

6. Run the `net.handle.apps.tools.SplitRecoveryLog` program to process the 'dbtxns.log' file. This will scan all of the handle modifications, creations, and deletions that have taken place since the backup/checkpoint operation. When this is finished, the date of the last transaction processed will be printed. Record this for future use.
7. *Only perform this step if you are splitting a primary server.* Shut down the old handle server. Copy the 'txn_id' file from the old server to each of the new server directories. This file will only be used by one of the new servers, but it is difficult to tell which one so we copy it to each server directory just to be safe.
8. Run the `net.handle.apps.tools.SplitRecoveryLog` program again, this time providing the date of the last transaction processed in Step 6 on the command line. (Run the `net.handle.apps.tools.SplitRecoveryLog` program with no arguments to see the syntax of the command.) This step will quickly bring the new databases into sync with the single primary by processing the transactions that have occurred on the primary since Step 6.
9. Start the new servers and make sure they come up without any problems.
10. Update the 'HS_SITE' value of the prefix or service handle with the new 'siteinfo.bin' file created in Step 2.
11. Wait at least 20 minutes so that the old 'HS_SITE' value is timed out in the caches of all possible handle clients/administrators.
12. *Only perform this step if you are splitting a primary server.* Copy the new 'siteinfo.bin' file into the directory of the old handle server. Start the old handle server back up. No administration requests should arrive. Performing this step should cause any secondary servers to notice the siteinfo version number change and retrieve the new siteinfo record from the old primary.

After some time has passed and everything is working OK, change the timeout/TTL of the modified HS_SITE value back to 86400 (one day).

2.9 Shutting Down a Handle Server

To stop a handle server, use the kill command for unix systems. Find the process id and then 'kill process_id'.

To restart the server, you may need to remove the lock file in your `svr/txns` directory if your server did not shut down cleanly. Then start the server using the command

```
jre -cp handle.jar net.handle.server.Main your_svr_dir
```

Please notify CNRI via email to hldadmin@cnri.reston.va.us if you plan to shut down your server permanently.

2.10 Inactive Prefixes

The Handle System Service Agreement requires Resolution Service Providers to ensure that the identifiers they create will resolve. If a handle service is shut down, the Handle System Administrator must be notified in advance, and arrangements must be made to enable clients that resolve handles homed to that discontinued service to correctly inform users of the status of those handles.

To help ensure proper resolution of such identifiers, the handle type HS_NAMESPACE is being proposed for use in alerting handle clients that a prefix is no longer supported or being used and handles created under that prefix have been removed. The Handle System Administrator would add an HS_NAMESPACE value type (a simple XML structure) to an inactive prefix to that includes status, contact information, and a customized message :

```
<namespace>  
<contact>clement@nus.edu.sg</contact>  
<status>inactive</status>  
<statusmsg>This prefix has been deactivated by the  
administrator as of September 2006.</statusmsg>  
</namespace>
```

Namespace-aware native handle clients (including the proxy server), that attempt to resolve handles under a prefix with an HS_NAMESPACE value indicating the prefix is inactive, will return a "Handle Not Found" error message including the contact and statusmsg data (if provided) stating that the prefix is no longer being maintained and all handles beginning with the prefix have been removed.

3 Handle Administration Tools

CNRI developed two GUI tools for performing handle operations such as creating and deleting handles, authenticating, and setting up a handle server. The tools have basically the same functionality, but the original "Admin Tool", in use for many years, has a few more options, including configuring a session and running backups, and is less "user-friendly" than the Handle Tool. The Handle Tool has been found to be useful for quickly creating and updating handles with URL values that are used with the World Wide Web.

Both tools are included in the HANDLE.NET software distribution. Section 3.1 describes the Admin Tool, and Section 3.2 describes the Handle Tool.

3.1 The Admin Tool

The Admin Tool is a graphical utility for performing handle operations. Before using this tool it is important that you read Section 2.6, *How Your Prefix is Set Up*.

Start the Admin Tool using the following command. (This assumes your 'handle.jar' file is in the '/hs/bin' directory.)

```
java -cp /hs/bin/handle.jar net.handle.apps.gui.hadmin.HandleTool
```

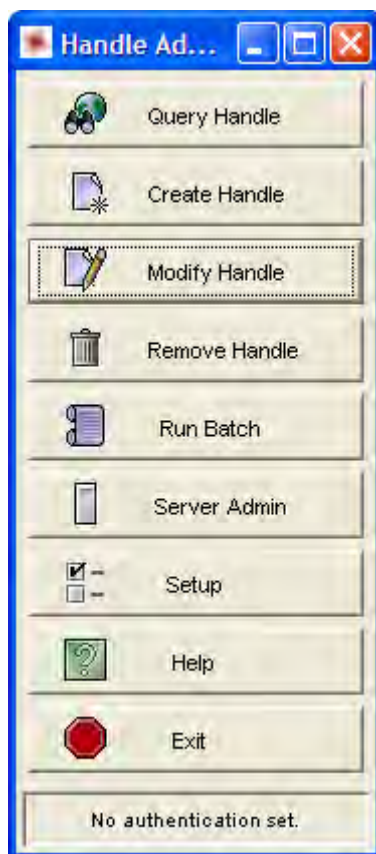


Figure 3.1.1 - Handle Admin Tool Main Menu

3.1.1 Query Handle

The 'Query Handle' button on the main Admin Tool window will display the window shown in Figure 3.1.2 below.

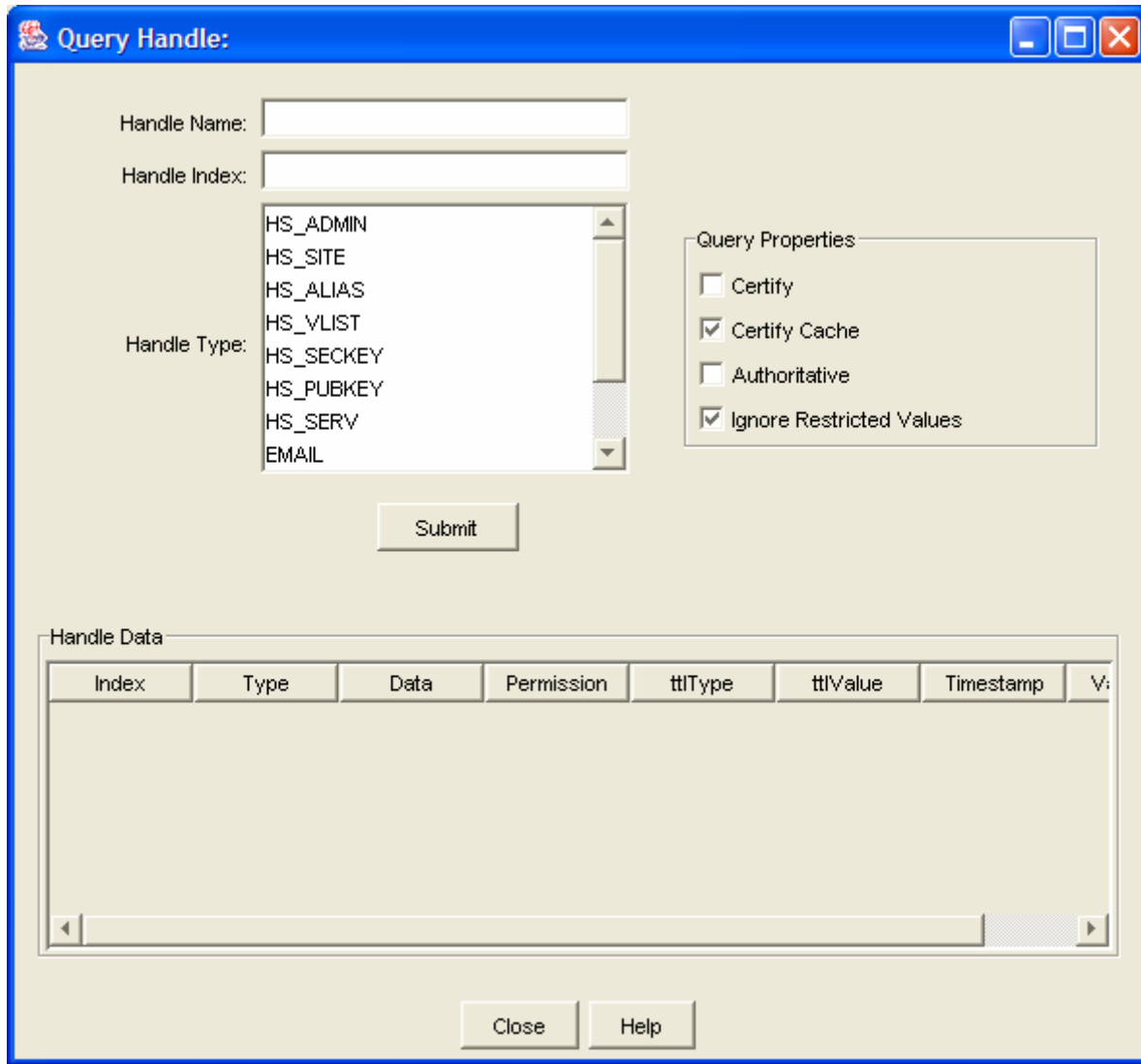


Figure 3.1.2 - Query Handle Window

Only authenticated users can query restricted handle values (non-public read). Other users can query public read handle values. Users can query specific types of handle values or specific index handle values.

3.1.1.1 Handle Name

Type a handle (prefix/suffix) in the 'Handle Name' text box.

3.1.1.2 Handle Index

Or, type the query indices of the handle values which you want to query in the 'Handle Index' text field. Use commas to separate multiple index values.

3.1.1.3 Query Properties

Or, users can select one, more or all handle value types to query by highlighting the types in the 'Handle Type' field.

Input Query Properties:

- Certify – indicates that the server must return the response message with the digital signature using the server's private key.
- Certify Cache – will require all cached values returned be signed by the originating server.
- Authoritative – do not use cache, always retrieve from responsible server.
- Ignore Restricted Values – Unauthenticated users check this to ignore nonpublic read handle values. Authenticated users can check this to retrieve nonpublic read handle values.

3.1.1.4 Submit

Press the 'Submit' button (or hit Enter) to process the query. A 'Resolving handle' window with a 'Cancel' button will pop up during processing. You can interrupt the query with the 'Cancel' button. An error message will pop up if the query failed.

3.1.1.5 Handle Data

The 'Handle Data' box displays the handle data values being queried. Highlight selected handle values from this list to display their content.

3.1.2 Create Handle

The 'Create Handle' button on the main Admin Tool window will display the window shown in Figure 3.1.3 below.

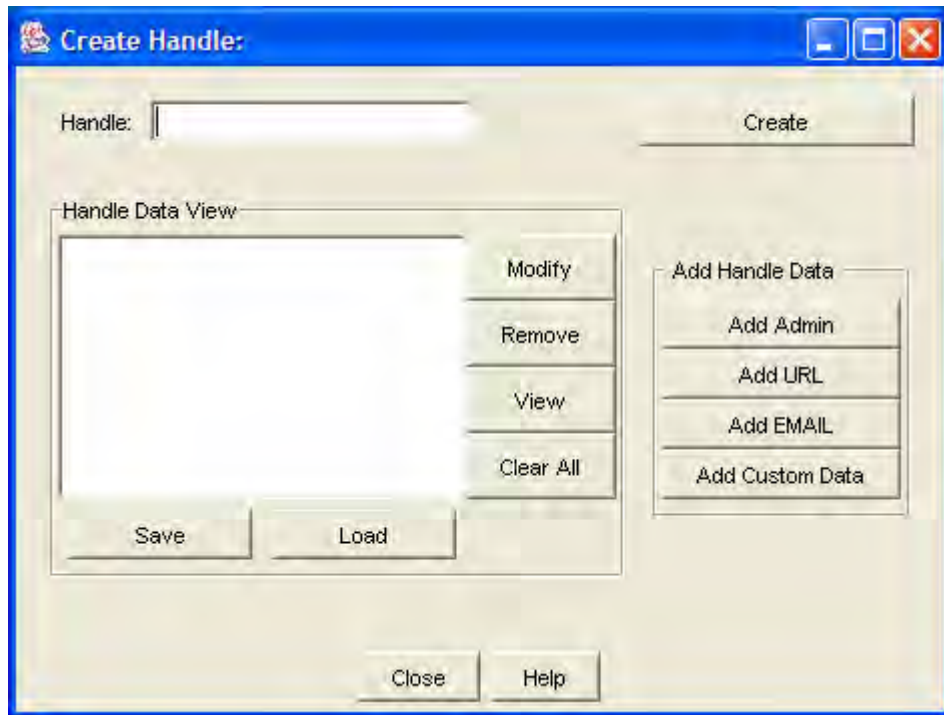


Figure 3.1.3 - Create Handle Window

Only authenticated users can create handles. Every handle **MUST** have at least one administrator. Every handle has a handle name and a group of handle values. Every handle value has an index, type, data, TTL (time to live), timestamp, permission set and references.

A handle has a set of values assigned to it and may be thought of as a record that consists of a group of fields. Each handle value must have a data type specified in its `<type>` field, that defines the syntax and semantics of its data, and a unique `<index>` value that distinguishes it from the other values of the set. A set of handle data types has been pre-defined for administrative use in the Interface Specification. (See RFC 3651, [Handle System Namespace and Service Definition](#), hdl:4263537/4068.)

`<type>` can be any UTF8-string. Handle System users acknowledge, however, that there are potential conflicts for handle clients if users assign types that are not registered and recognized across the user community. How `<types>` should be defined and how they should be used is currently under discussion. The non-administrative types that have been registered and defined to date are listed below.

- **URL:** Values of type URL are UTF8-encoded URIs that specify the location of the object identified by a handle.
- **EMAIL:** Values of type EMAIL are UTF8-encoded email addresses.
- **DESC:** Values of type DESC are UTF8-encoded text descriptions of the object identified by the handle.

3.1.2.1 New Handle

If the user has not previously authenticated himself by selecting 'Setup/Authentication' from the main Handle Admin Tool menu, it should be done now. (See also Section 3.1.10, *Authentication*.) To change the authentication information between handle creations, return to 'Setup' on the main menu, then click 'Authentication'. Type a new handle (prefix/suffix) in the 'Handle' text box. Press 'Return'.

3.1.2.2 Add Handle Data

The 'Add Handle Data' box contains shortcut buttons for quick addition of certain handle types with defaults already set. The 'Add Custom' box is used to add handle values with custom types. Every handle value must have an index to identify it within the handle record.

Add Admin: used to create an administrator for the new handle. Every handle **MUST** have at least one administrator. Every administrator value is made up of a handle and handle value index. The 'Admin ID Index' can be any unique number within the handle data for the handle being created. The 'Admin ID Handle' and 'Admin ID Index' identify either:

A public key or secret key that an administrator must authenticate against.

An admin group that references (possibly indirectly through multiple admin groups) a handle value with either a public key or secret key that an administrator must authenticate against.

Be sure to check the appropriate permissions for the administrator handle. The 'More' button to view or modify the Type, TTL, timestamp, permissions, and references related to this handle value.

Add URL: used for the creation of a URL data type for the new handle. Click the 'More' button to view or modify the Type, TTL, timestamp, permissions, and references related to this handle value.

Add Email: used for the creation of an email address for the new handle.

Note: The Add Handle Data windows each have a 'More' button which can be used to view or modify the Type, TTL, timestamp, permissions, and references related to each handle value.

Add Custom Data: used to create additional data values not shown on the 'Create Handle' window and for the customization of permissions and TTL values. This button will bring up the window shown in Figure 3.1.4 below:

Figure 3.1.4: Input Custom Info Window

Select a type from the pulldown menu (HS_ADMIN, HS_SITE, HS_ALIAS, HS_VLIST, HS_SECKEY, HS_PUBKEY, HS_SERV, EMAIL, URL, URN, INET_HOST) in the 'Type' box or input a new one. The 'Value Data' button inputs the data corresponding to the type.

Note that the type **DESC** has been registered, but does not yet appear in the pull down list in the Admin Tool. Values of type DESC are UTF8 encoded text descriptions of the object identified by the handle.

Also, the type URN has not been registered or defined, and although it appears in the list, its use is discouraged.

Handle types are registered as handle values. For example, if you use the proxy server to resolve the URL <http://hdl.handle.net/0.TYPE/DESC> you can see the handle values and the full description of the type DESC.

HS_ADMIN type data a new administrator (See **Add Admin** section above).

HS_SITE type data adds the site information for prefixes in the Global Handle Registry to indicate where handles with that prefix are resolved. (HS_SITE is used only by the CNRI Handle Administrator, for prefix handles.) The data value must have an index value which can be any unique number within the handle record data. The data version, protocol and serial number are values that have to do with the current Handle System version. Check whether the site is a primary or a multi primary. Choose whether the handle will be hashed by the entire handle, just the prefix, or a local name. Add the IP addresses of the servers that exist in the site. Add attribute value pairs.

HS_ALIAS type data is used to add a handle alias as a handle value.

HS_VLIST type data is used to define administrator groups with a list of other handle values.

HS_SECKEY type data adds a secret key as a handle value. Generally, you should check the 'public read permission'.

HS_PUBKEY type data adds a public key as handle value. Generate key pairs (private key, public key) using the 'Generate Key Pair' button. Load a public key from the file system using the 'Load Key' button then add the public key to the key field. Click the 'Clear' button to clear the key field. Click the 'Ok' button to confirm.

HS_SERV type data is a handle value which has site information for a service (for prefixes only).

EMAIL type data is a handle value which stores an email address.

URL type data is a handle value which stores a URL.

URN type data is a handle value which stores a URN. *Note: This type appears in the Admin Tool's list, but it has not been defined or registered and its use is undergoing review.*

INET_HOST type data is a handle value which stores an IP address or host name.

3.1.2.3 Handle Data View

This box displays the handle data values being added. The 'Modify' button allows you to change the selected handle value. The 'Remove' button allows you to remove the selected handle value. The 'View' button allows you to view the selected handle value. The 'Clear All' button allows you to remove all handle values

3.1.2.4 Save and Load

The 'Save' button allows you to save the handle values to a file. The 'Load' button allows you to load the handle values from a file, and append those values to the current handle value group or reset them as the current handle value group.

3.1.2.5 Submit

To submit the created handle press the 'Create' button after the addition of all the handle values is complete. The response will indicate success or failure. The absence of an administrator and a handle name is considered a failure.

3.1.3 Modify Handle

The 'Modify Handle' button on the main Admin Tool window will display a window that looks much like the handle creation window. To modify a handle, first type it into the text box at the top of the window. Then hit the 'ENTER' key to retrieve the handle's current values. You can then use the modify, remove, and add buttons.

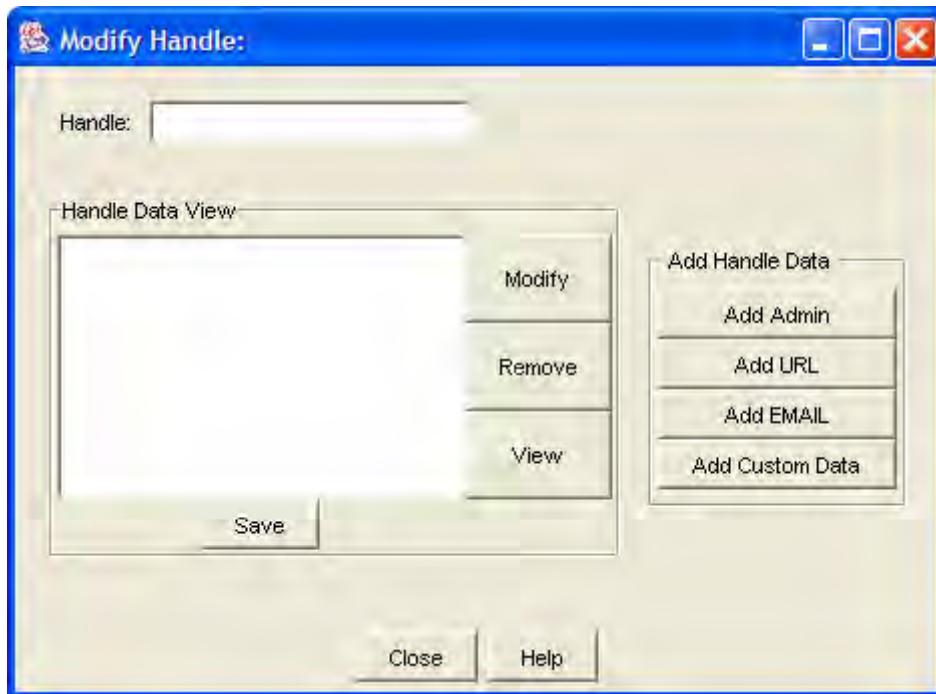


Figure 3.1.5 - Modify Handle Window

It is important to note that the handle will be modified after each operation. If you plan on replacing an admin value, you should always add the new value first, then remove the old.

3.1.4 Remove Handle

The 'Remove Handle' button on the main Admin Tool window will display a simple window for handle deletion. Enter in the handle to remove into the text box at the top of the window. Then hit the 'ENTER' key to view the handle's current values. Finally, click the 'Remove' button to delete the handle.

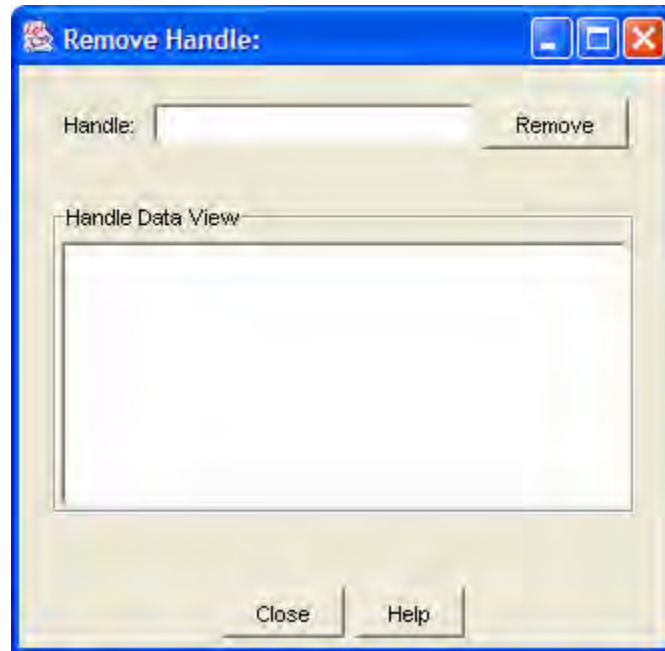


Figure 3.1.6 - Remove Handle Window

3.1.5 Running Batch Files

The 'Run Batch' button on the main Admin Tool window will display a window for submitting batch files.



Figure 3.1.7 - Batch Submit Handles Window

Only authenticated users can submit batch files. Batch files need to follow the file format described in the Chapter 4 *Batch Operation*. Batch files can include more than one kind of handle operation (CREATE, DELETE, ADD, REMOVE, MODIFY, HOME, UNHOME). Users can authenticate themselves either through the batch files or through the administrative tools.

3.1.5.1 Load Batch file

Click the 'Add' button to enter the batch file path. This will be added to the batch file list window. Click the 'Modify' button to change the selected file's path. Click the 'Remove' button to delete the selected file's path from the list. Click the 'View' button to view the selected batch file's path fully without editing. Click the 'Clear All' button to delete all files from list.

3.1.5.2 Authenticate

There are 2 ways to authenticate:

1. Select 'Setup' from the main Handle Admin Tool menu, then click 'Authentication'.
2. Include authentication information in the batch file as shown in Section 4.1 *Create Handle Batch Format*.

3.1.5.3 Batch Submission Log

There will be output from the batch submission. Select the corresponding radio button to output the log information to a specified file, to stdout, or to the log window. If you chose to output the log to a file, enter the log file path. There are three types of log messages:

- 'SUCCESS' means the operation completed successfully.
- 'FAILURE' means the operation failed.
- 'INVALID' means the format of the operation was invalid.

3.1.5.4 Submit Batch

Click the 'Submit Batch' button to submit the batch operation. If you want to interrupt the batch submission process, click the 'Stop Batch' button.

3.1.6 "Home" a Prefix

"Homing" a prefix on a particular site tells the server(s) that make up the site that they are responsible for the given prefix. This way, when a resolver comes along and asks for a handle under that prefix, the server can say "Here it is." or "It doesn't exist." or even "Why are you asking me? I don't have it."

Permission to Home and Unhome is in the server admins parameter in the `config.dct` file in the server directory. Click on the 'Server Admin' button to select "Home Naming Authority". If you enter the prefix handle as well as the address and port number of one of the primary servers for the desired site, this tool will "home" the given prefix to that site. A message will be sent to each server in the site indicating that that site will now be responsible for the given prefix. From then on that server will accept all requests for the given prefix.

3.1.7 "Unhome" a Prefix

"Unhoming" a prefix on a particular site tells the server(s) that make up the site that they are no longer responsible for the given prefix.

Click on the 'Server Admin' button to select "Unhome Naming Authority". If you enter the prefix as well as the address and port number of one of the primary servers for the desired site, this tool will "unhome" the given prefix on that site. A message will be sent to each server in the site indicating that that site will no longer be responsible for the given prefix. From then on that server will reject requests for the handles under the given prefix.

3.1.8 Backing Up a Handle Server

The Backup Server function of the admin tool sends a request to a server to checkpoint its internal handle database. (Click on the 'Server Admin' button to select "Backup Server".) In order to be able to checkpoint a server, the administrator must be identified as an administrator for that server (in the `backup_admins` section of the `config.dct` file on the server).

The checkpoint operation consists of several steps. Upon receiving an authenticated request to backup the database, the server will

1. Copy the main database files ('`handles.jdb`' and '`nas.jdb`') to backup files ('`handles.bak`' and '`nas.bak`')
2. Reset the transaction log ('`dbtxns.log`')

After these steps the '`handles.bak`' and '`nas.bak`' files can be safely copied to another location for a backup. The '`dbtxns.log`' file will contain all of the changes made to the database since the '`handles.bak`' and '`nas.bak`' files were made. The '`dbtxns.log`' file will allow you to restore the backup up to the last transaction that was successfully performed if something were to go wrong with the main database.

To perform the checkpointing, click the 'Server Admin' button and select 'Backup Server'. Enter the IP address and port number of the server that you want to perform the checkpoint operation.

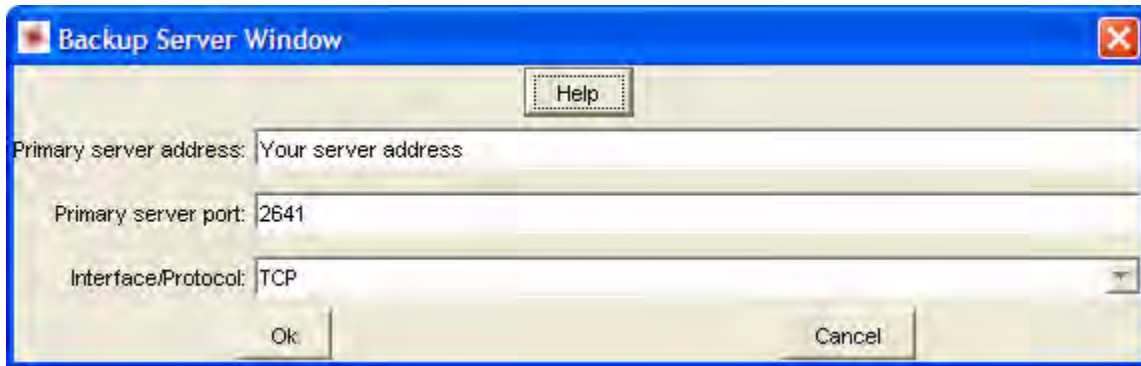


Figure 3.1.8 - Backup Server Window

During the checkpoint process, the server will reject all requests to create, modify, or delete handles. For this reason, it is usually preferable to perform the checkpoint operation when there is little administrative activity on the server. Checkpoint operations should only be performed on primary servers since secondary servers do not keep transaction logs for their databases.

To recover the database using the backup files and transaction log you can perform the following steps:

- 1 Make sure that the server is NOT running.
- 2 Make extra copies of all files (doesn't hurt to be safe!)
- 3 Run the command:

```
java -cp handle.jar net.handle.apps.tools.RecoverJDB  
<server_dir>
```

- 4 Restart the server. The server should now have its database restored to its pre-disaster state.

3.1.9 Listing Handles on a HANDLE.NET Server

The "List Handles" function of the admin tool sends a request to a service to list all of the handles for a specific prefix. In order to be able to list handles the administrator must have the "List Handles" permission enabled in the prefix.

Click on the 'Server Admin' button to select "List Handles". The List Handles function is implemented in the most recent CNRI handle server, but if your handle database is very large, the list handles command may timeout since the database used by the CNRI handle server is not optimized for this kind of operation.

3.1.10 Authentication

In order to authenticate, the user will need a handle value and its associated secret key or private key. To authenticate, select 'Setup' from the main Handle Administration Tool menu. Select 'Authentication'. Select the Authentication Type of 'Secret Key' or 'Public Key'. Enter the 'ID Handle' and 'ID Index'. Enter the 'Private Key File' path or the 'Secret Key'. Click 'OK' to begin authentication or click 'Cancel' to cancel it.

3.1.11 Generate Key Pairs

This window will enable the generation of a public key pair. (Please reference RFC 3651, [Handle System Namespace and Service Definition](#), Chapter 5.2 *Handle System Authentication Protocol*, and [System Fundamentals: Authentication](#) on the Handle System web site, for more information.

To generate key pairs, follow these steps:

1. Select 'Setup' from the main Handle Administration Tool menu.
2. Select 'Generate Key Pairs'.
3. Enter the paths of the private and public key files in the corresponding text fields or use the 'Browse' button to find the files.
4. Select the Algorithm to be used.

5. Enter the Strength of the key pair to be generated. The key length is variable from 512 to 1024 bits. The default is 1024 bits. The longer the length, the stronger the key pairs.
6. Select 'Encrypt' or 'Do not encrypt' of the private key. Encryption of the private key requires that you choose a secret passphrase that will need to be entered whenever authenticating using this key pair. (Note that if you choose not to encrypt your key, and later change your mind, use KeyUtil.java (in the distribution in /src/net/handle/apps/tools/) to encrypt your existing key, and then send the encrypted file to the Handle System Administrator.)
7. Click 'GenKeys'.
8. If 'Encrypt' was selected, a window will prompt you to enter your secret passphrase.
9. A message will confirm the generation of the keys. Click 'Close' to exit the 'Generate Key Pair' window.

3.1.12 Using Sessions

Sessions reduce the authentication processing time for performing a sequence of administrative operations. Sessions also enable the encryption of transactions between the client and hosting server.

Authenticated users establish a session with a server by selecting the desired session mode from the "Session Mode" list and setting session attributes. Each user explicitly sets session setup options via this panel. The Session Setup panel is displayed in Figure 3.1.19 below.

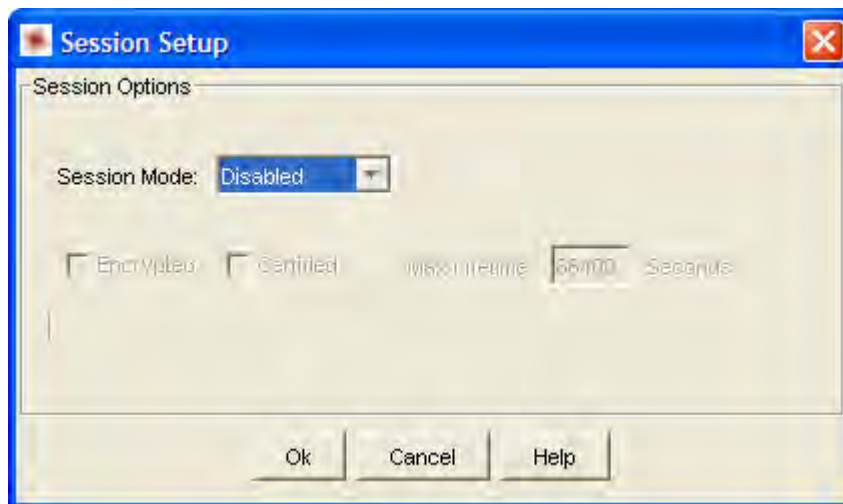


Figure 3.1.9: Session Setup Window

To enable sessions, follow these steps:

1. Choose a session mode. *Note that not all modes may be available.*
 - A. Choosing a session mode of "disabled" will turn off sessions. The Admin Tool will use the standard method of communication with the server.

- B. Choosing a session mode of "DiffieHellman" will use a public key pair supplied by the Admin tool. This is the simplest mode to configure.
- C. Choosing a session mode of "client cipher" will use a public key pair supplied by the client. You must specify files for a public and private key pair in order to use this mode. (not available yet).
- D. Choosing a session mode of "cipher reference" will use a public key stored in a handle. You must specify the handle that contains this public key, the index at which the key is stored, and a file that contains the corresponding private key in order to use this mode. (not available yet)

2. Specify session options:

- A. Encrypted allows you to specify that all session messages from the server must be encrypted using the session key.
- B. Certified allows you to specify that all session messages from the server must be certified with a MAC code using the session key.
- C. Max lifetime allows you to specify a session time out in minutes. The default sets a server session timeout of 24 hours (3600 minutes). Invalid entries in this field are zero and any negative number. Very large numbers will not validate, resulting in the handle software using the default timeout or your previous setting.
- D. Click the OK button to save your session setup information. The new settings will take effect when your next administrative operation is processed.

3.1.13 Console

The Console Window displays debugging messages concerning requests sent to a handle server such as resolution. This is useful to see where server packets are sent in the process of handle administration and/or resolution.

To display the console, select 'Setup' from the main Handle Administration Tool menu and select 'Show Console'.

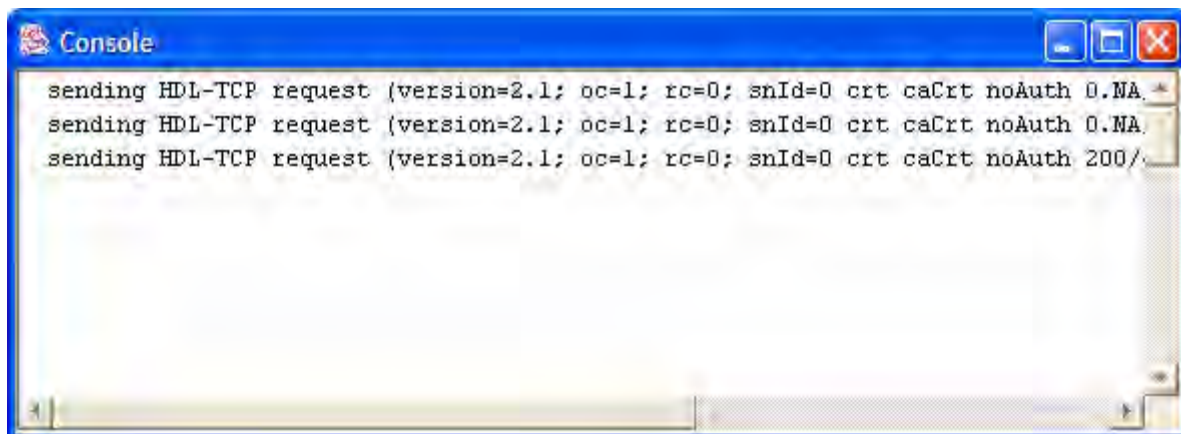


Figure 3.1.10 - Console Window

3.2 The Handle Tool

The Handle Tool, found by some users to be simpler to use than the original Admin Tool, runs on Windows, Linux and Mac OS X, and requires that Java™ 1.5 be installed. The tool uses Java™ WebStart, which is built-in on OS X and comes with Java™ 1.5 for Windows and Linux.

3.2.1 Installation

Download and install the Handle Tool:

For Windows:

- 1) Install Java™ 5 (j2se 1.5) or higher from <http://java.com/>.
- 2) Click the following link: <http://www.handle.net/webstart/hdlgui/hdlgui.jnlp>.
- 3) Double-click the `hdlgui.jnlp` file to download and run the application.
- 4) Click the accept/start button when asked to trust the application.

For Linux:

- 1) Install Java™ 2 or 5 (j2se 1.4 or higher) from <http://java.com/>.
- 2) Run the following command:

```
javaws http://www.handle.net/webstart/hdlgui/hdlgui.jnlp
```

- 3) Click the accept/start button when asked to trust the application.

For Mac OS X (comes with Java™ already installed):

- 1) Click this link: <http://www.handle.net/webstart/hdlgui/hdlgui.jnlp>.
- 2) Double-click the downloaded `hdlgui.jnlp` file to run the application.
- 3) Click the accept/start button when asked to trust the application.

After running the application for the second time, you will be prompted to install the application on your desktop. We suggest doing so.

3.2.2 Default Window

The application's Default Window, displayed at launch, is a panel used to look up an existing handle's values; to create, modify or delete handles; and to authenticate an administrator.

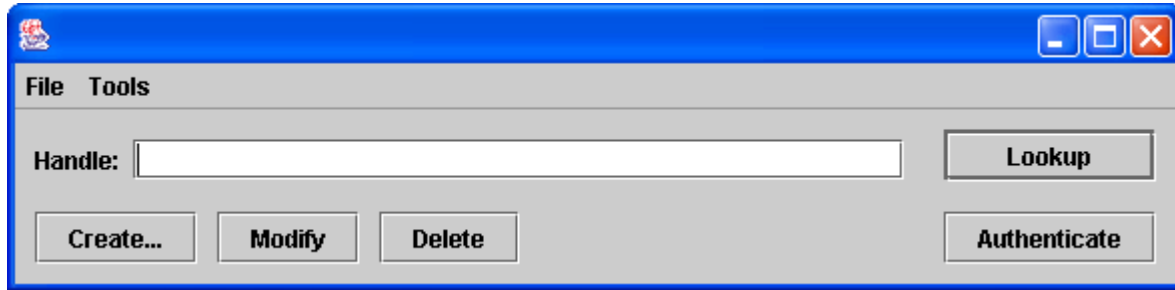


Figure 3.2.1: Default Window

Default Window Options

Under the **File** menu in the default window there are four options (see Figure 3.2.2):

- (1) **Open Batch File.** Open an existing batch file stored on the desktop, prepared according to the specification and ready for upload. The pathname of the file will be added to the Batch Process Tool. (See Section 3.2.1.2, **Batch Processing**.)
- (2) **Clear Cached Handles.** Empty the cache of data stored from previous handle resolutions.

Like most name resolution systems, the Handle System makes use of caching (storing the values it retrieves from the Handle System) in order to improve performance and keep network traffic and server load to a minimum. The default configuration for the Handle Admin Tool is to cache all handles as they are resolved, and to use the cached values whenever possible. This allows the tool to display the resolved values of most handles with little or no delay. However, if you know that a handle has been modified, but the change is not reflected in the View Handle display, click **Clear Cached Handles**. The next time that handle, or any handle, is resolved, it will be retrieved from the handle server rather than the local cache. The cache is automatically cleared each time you quit the Handle Tool.

(See also Section 3.2.6.3, **Add... Button** for more information on caching, TTL, and TTL Type.)

- (3) **Authenticate.** Authenticate as an administrator using Public/Private Key pair or a Password (Secret Key). See Section 3.2.4, **Authentication**.

- (4) **Quit** the Handle Tool.

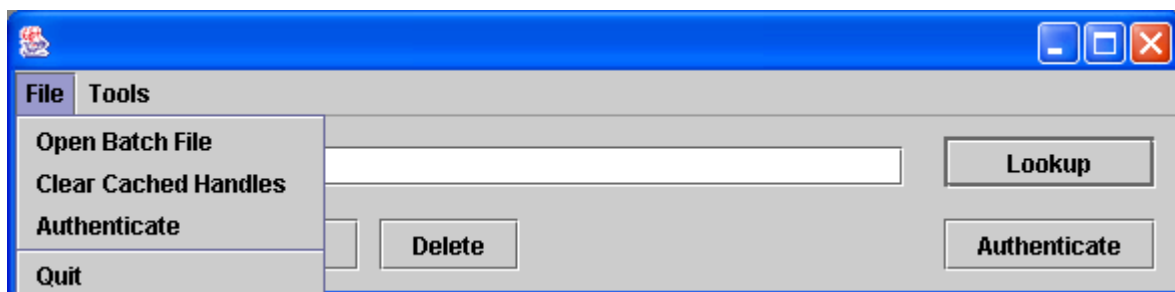


Figure 3.2.2: The File Menu

Under the **Tools** menu in the default window there are three options (see Figure 3.2.3):

- (1) **Batch Processor.** Uploads handle records contained in a batch file. (See also Section 3.2.10, *Batch Processing*.)
- (2) **Console.** Displays the communication between the computer you are on and the handle servers.
- (3) **Home/Unhome Prefix.** Tells servers in a site that they are (or no longer are) responsible for resolving handles with a given prefix. (See Section 3.2.1.3 *Homing/Unhoming a Prefix*.)

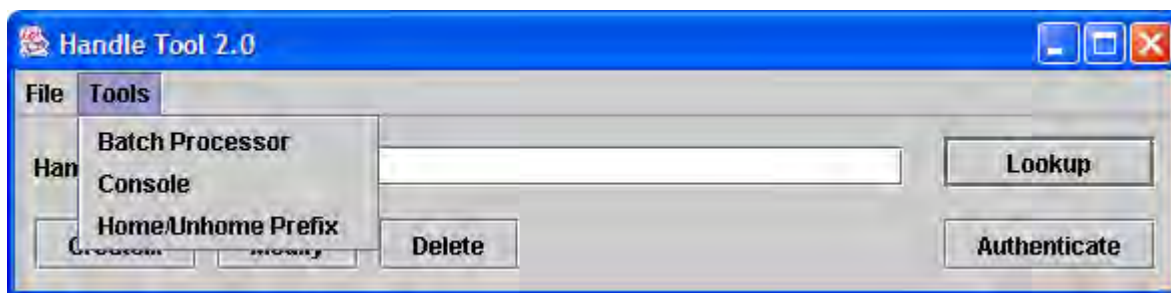


Figure 3.2.3: The Tools Menu

3.2.3 Console

The console records the interaction between the Handle Tool and the handle servers it contacts.

Using the Console Window

The **Console** window can be opened at the time the Handle Tool is launched, and moved to the side of the desktop (see Figure 3.2.4). Use the Console to view the communication between the computer you are using and the Handle System. You will also see general diagnostic and log messages displayed in the console.

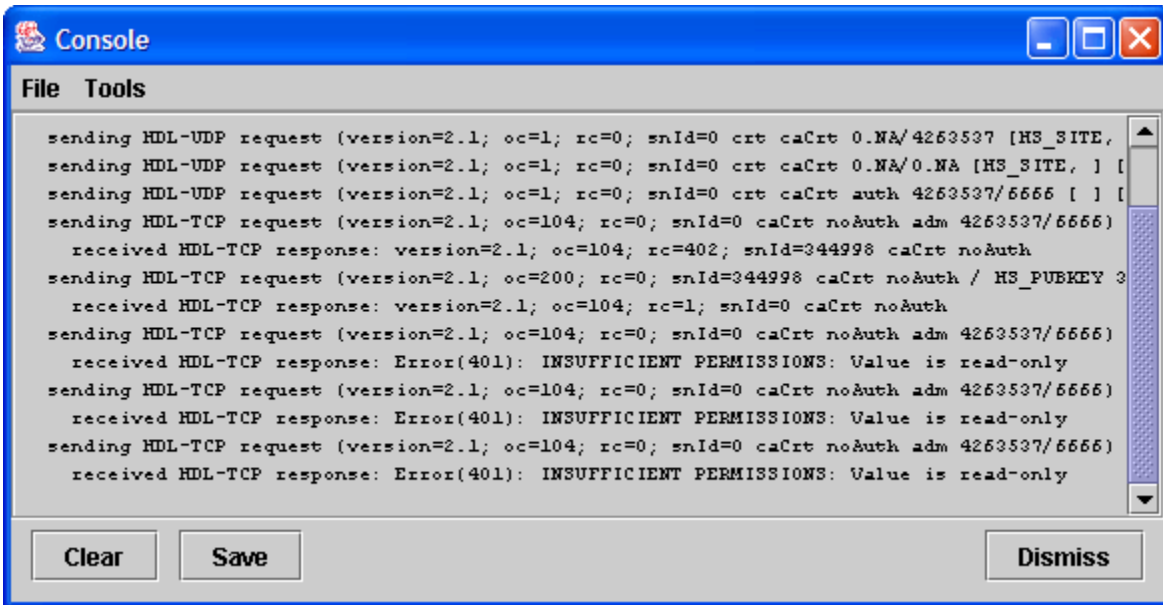


Figure 3.2.4: Console

The Console displays all client requests sent to the server (including the IP address of the server) and all responses received from the server, and will maintain the list until the Handle Tool is quit. The contents of the window can be cleared at any time by clicking the **Clear** button, or saved to a desktop file for analysis by clicking the **Save** button. Error messages should be saved for troubleshooting. **Dismiss** closes the window.

3.2.4 Authentication

You must have an administrative handle to use for authentication in order to administer handles. When authenticating yourself you will be providing the following information:

- (1) Your admin handle (which is usually your prefix handle) and the index of your public key value within that handle; or, some organizations use individual handles with passwords.
- (2) Your private key. Note: your private and secret key will **never** be sent over the Internet by the Handle System. You only need to provide this to the Handle System client software so that it can prove to any handle server that you have this information.

The owner of a prefix under which you are administering handles (the part of the handle before the slash) will have given you permission to create handles under that prefix by adding your administrator handle, and the index for your key value, to a list of administrators who have permission to create handles under that prefix.

When you send a "create handle" request to the Handle System, the server will verify that you are the individual identified by the administrator handle (i.e., your private key matches your public key, or you enter the correct password (secret key)) before the requested handle is created.

Administrator Authentication

Select **Authenticate** from the **File** menu, or click **Authenticate** on the Default Window to display the Handle Authentication Window (see Figure 3.2.5 below).

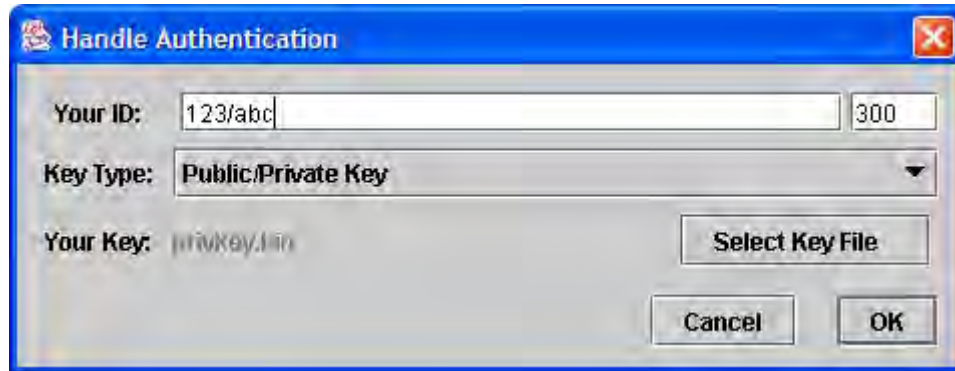


Figure 3.2.5: Handle Authentication Window Showing Public/Private Key Authentication

Enter your administrator handle into **Your ID:**. The default index value is 300. (Change this value only if required by your system administrator.)

Select the **Key Type:**, either Public/Private Key as illustrated above, or Password (Secret Key).

Your Key: will display the name of your Key file. If you are using the tool for the first time, or your private key file has moved or changed, click **Select Key File** to browse for the (updated) file.

Click OK. If you use Public/Private Key, you will next be asked to enter your passphrase (see Figure 3.2.6 below).

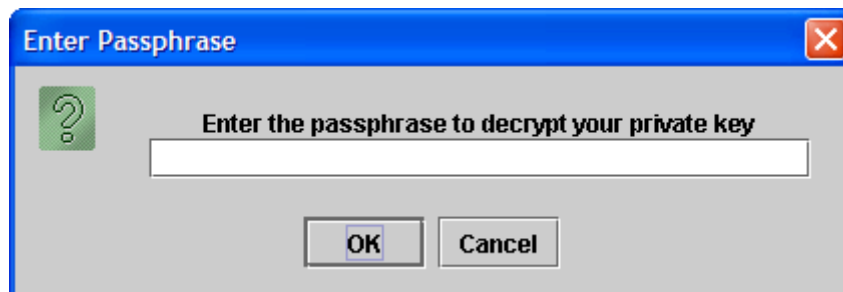


Figure 3.2.6: Enter Passphrase

If you selected Password, the Handle Authentication window will allow you to enter your password (see Figure 3.2.7 below).

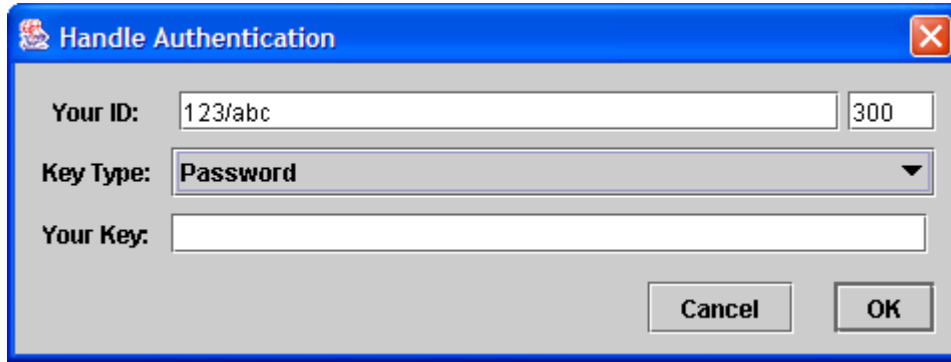


Figure 3.2.7: Handle Authentication Window Showing Password (Secret Key) Authentication

When authentication is successful, your administrator handle will display in place of the Authenticate button on the default window (see Figure 3.2.8).

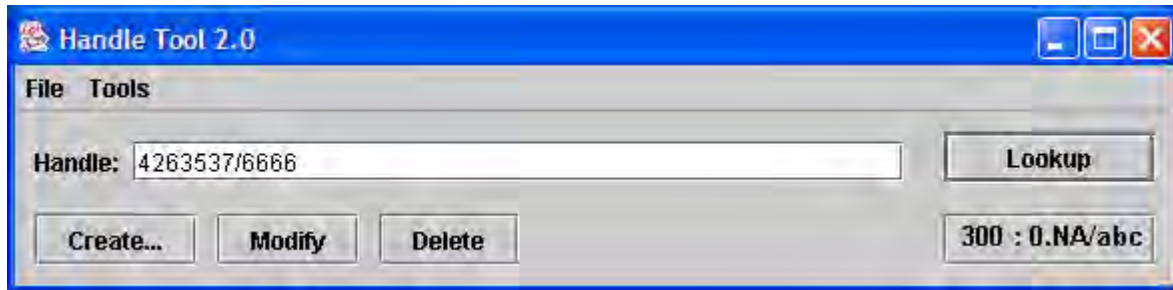


Figure 3.2.8: Authentication Complete

3.2.5 Lookup

It is not necessary to authenticate yourself as a handle administrator before viewing a handle record. Some low level system handle values remain hidden from view, but values that have public read permission and might need to be edited by an administrator will be displayed in this view.

3.2.5.1 Lookup a Handle Value

Enter the handle into the Default Window's textbox, and click **Lookup**. The publicly readable values stored in the handle will be displayed in the View Handle window (see Figure 3.2.9 below).

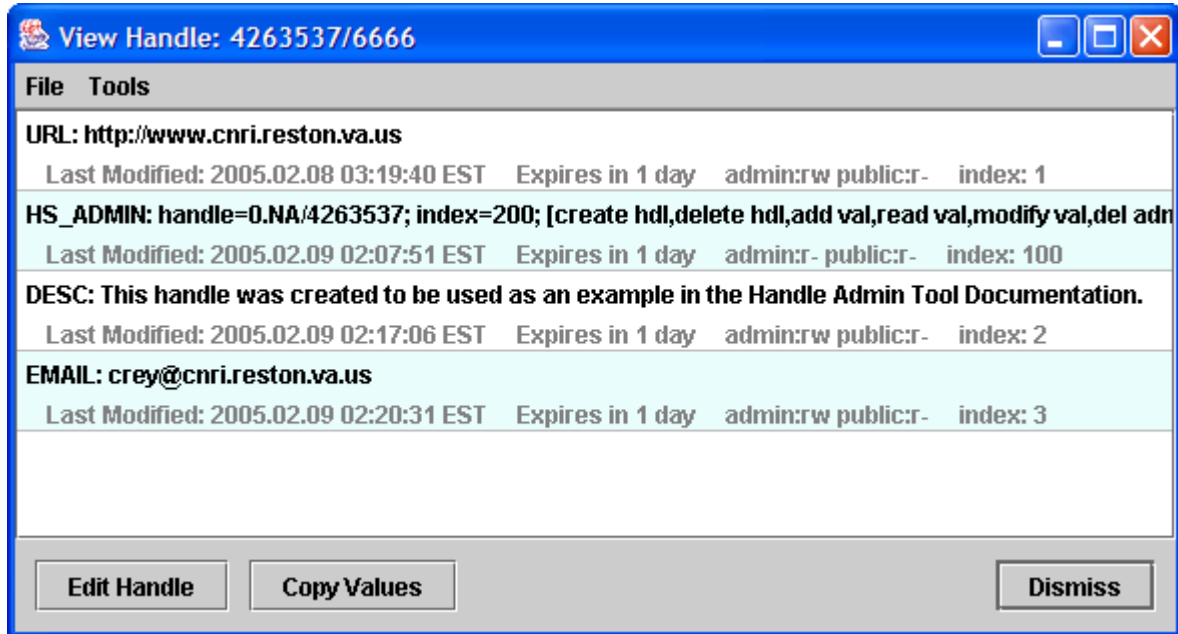


Figure 3.2.9: View Handle Window

3.2.5.2 Edit Handle

To edit one or more of the values for the handle displayed in the View Handle window, click **Edit Handle** to bring up the Edit Handle Window. (See Section 3.2.7 **Modify** for instructions for Modifying a handle using the Edit Handle window.)

3.2.5.3 Copy Values

The Copy Values function is a short cut for creating handles with the same values as an existing handle.

To create a new handle that will have all (or most) of the same values as the handle being displayed in the View Handle window, click **Copy Values**. Type the new handle string into the window, and the new handle will be created with all of the copied values. The new handle values will be displayed in a Create Handle window, which includes a button for removing any unwanted values.



Figure 3.2.10: Copy Values to New Handle

3.2.6 Create, Modify or Delete Handles

Handles can be created, modified or deleted individually or in batches. This section explains how to create, modify or delete handles one at a time. See Section 3.2.10 **Batch Processing** for information on creating and processing batches.

If you do not authenticate yourself before attempting a create/modify/delete, the Handle Authentication window will appear as the first step. (See Section 3.2.4, **Authentication**, for instructions.) You must have permission to perform the requested action on any given handle.

3.2.6.1 Create New Handle

To create a new handle, use the **Create** button on the Default Window. Handles may have any number of associated values. By default, the Handle Tool will assign an administration value to the new handle. See Section 3.2.6.4 **Value Types** for an explanation of the types of values that can be included.

If you are creating a handle that contains the same (or most of the same) values as an existing handle, do a Lookup for the existing handle and click **Copy Values** on the View Handle window (see Section 3.2.5.3 **Copy Values**).

Administration Values

Every handle must have at least one "administration value" associated with it, to identify the administrators that have permission to act on that handle, and which permissions each administrator will have. Administration (or admin) values are values of type HS_ADMIN, and for consistency are being given an index of 100. (If there are multiple admin values, then the additional indexes are 101, 102, 103, and so on.) HS_ADMIN values specify who can perform administration by referring to values stored in other handles.

3.2.6.2 Create Handle Steps

To create a handle, first enter the new handle string into the text box on the Default Window, and then click **Create**.

The **Create** button offers three options (see Figure 3.2.11 below).

- (1) Selecting **Create Simple URL Handle** will display a panel for quick entry of a handle's URL value, and then display that value in the Create Handle window.
- (2) **Create Simple Email Handle** will display a panel in which to enter an email address, and then display that value in the Create Handle window.
- (3) **Create Blank Handle** will bring up the default Create Handle window (see Figure 3.2.11 below).

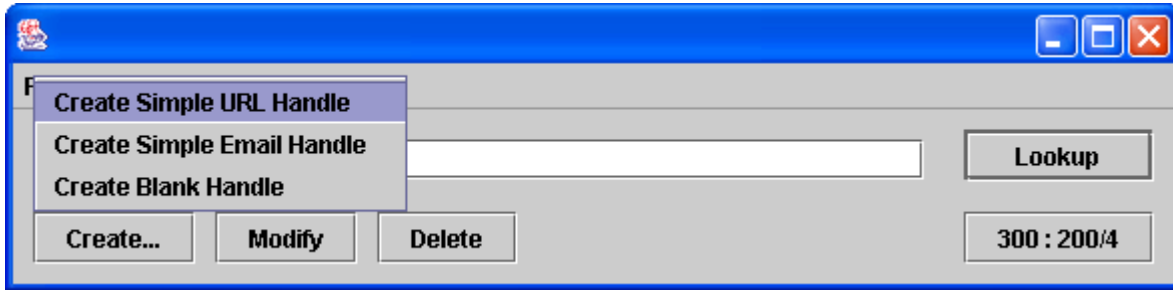


Figure 3.2.11: Create Button Options

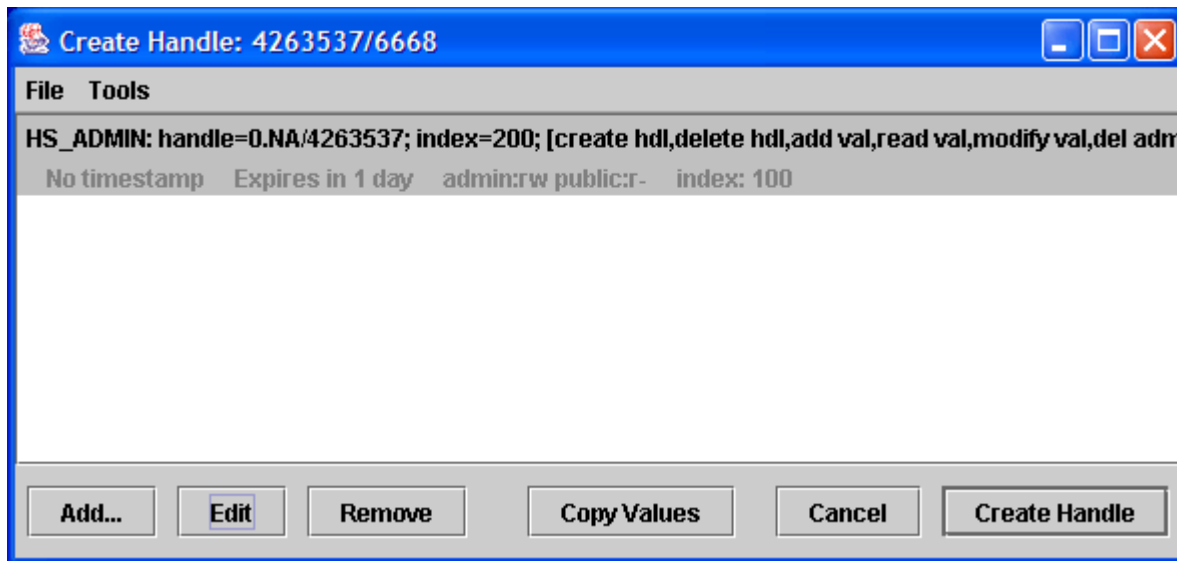


Figure 3.2.12: Create Handle Window

The new handle will automatically have a default HS_ADMIN value that, depending on the configuration of your naming authorities, will most likely be sufficient. Additional HS_ADMIN values can be added.

3.2.6.3 Add... Button

To add additional values to the handle record, click **Add**.

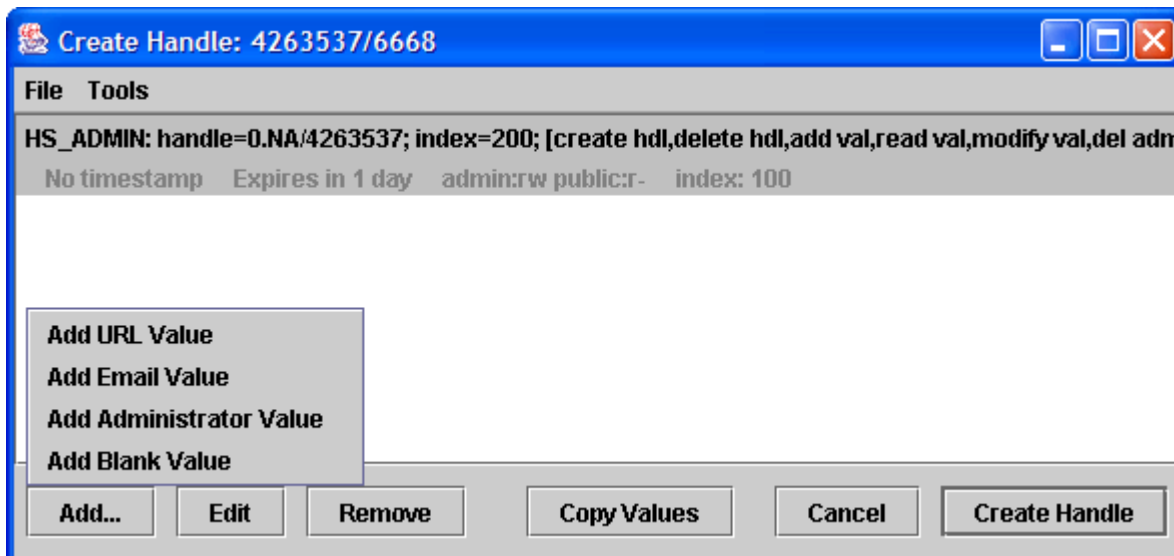


Figure 3.2.13: Add... Button Options

Selecting one of the **Add** button options affects the Value Type displayed in the Add Value window. Selecting URL, Email or Administrator will fill in the appropriate Value Type in the Add Value window. **Add Blank Value** requires the administrator to identify the type for the new value.

Note: If you have selected **Create Simple URL Handle** and entered a URL into the text box, you need only click **Create Handle** to complete the process and create the handle. You will not need to add additional values as explained in the remainder of this section. When the handle has been created, a message as shown in Figure 3.2.14 will display.

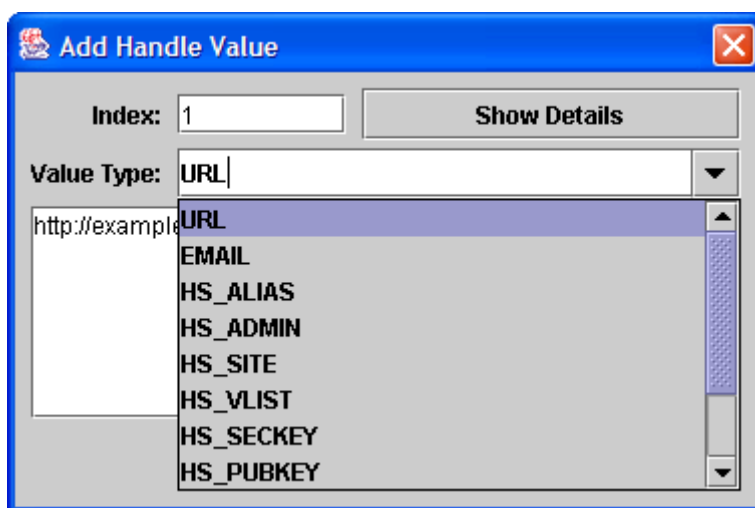


Figure 3.2.14: Add Handle Value

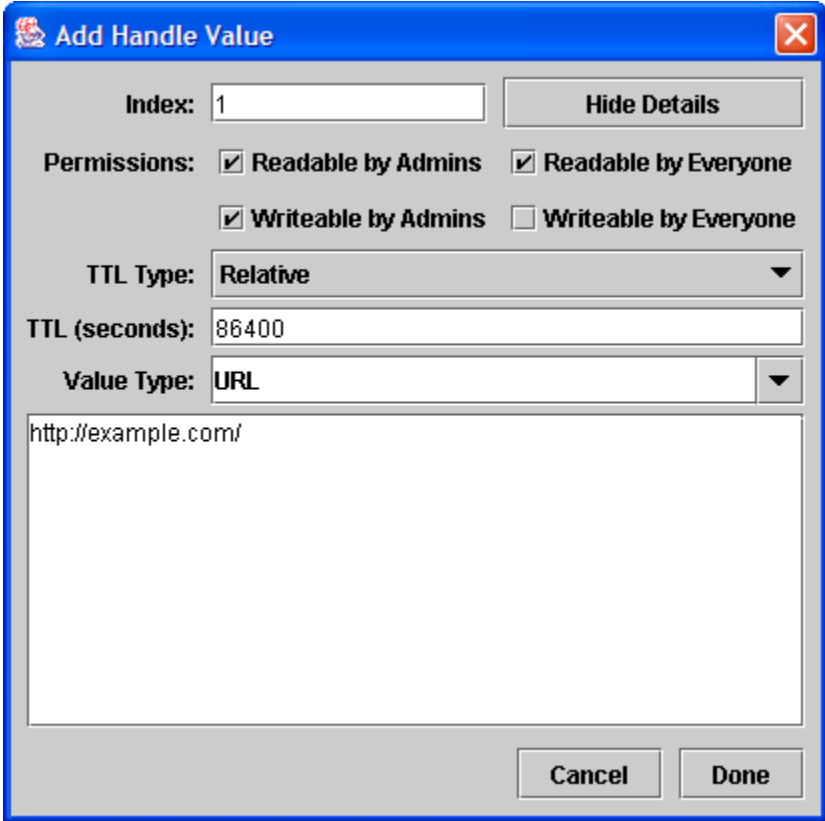
For example, selecting Add URL Value will fill in the Value Type with URL. (Selecting Add Blank Value will leave the Value Type blank so that the appropriate type can be selected from the pick list under the down arrow, shown in Figure 3.2.14 above.) The URL is added in the text window, in place of the example URL shown.

Clicking **Show Details** will display additional data about the new value being added (see Figure 3.2.15). It is only necessary to use Show Details when changes to the default settings are required. The default set of permissions is the recommended practice for most handles. ("Writable by Everyone" permission is not recommended.) Should it be necessary to change the **Index** value or **TTL** (Time To Live) values that affect caching, use this window to change the settings.

TTL stands for "Time to Live" which indicates that maximum number of seconds a handle value can be cached before a handle client must clear the value and retrieve it again from the Handle System. Setting the TTL to a low value forces handle clients to retrieve the handle more often, so low values are more appropriate for handles that change frequently.

The TTL Type provides a choice between "relative" and "absolute". If the TTL Type is "relative" then the TTL value will indicate the number of seconds since being retrieved that the handle value is valid. If the TTL Type is "absolute" then the TTL value indicates the number of seconds after Jan 1st, 1970 12:00 am GMT that the handle value is valid. The absolute TTL Type is useful for handle values that expire or will be updated at a specific date and time.

The default TTL Type is "relative" and the default TTL is 86400 which means that the handle value can be cached for only 24 hours before it should be re-retrieved. There are very few situations in which the default settings are not appropriate.



The screenshot shows a dialog box titled "Add Handle Value" with a blue title bar and a close button. The dialog is divided into several sections. At the top left, there is a text box labeled "Index:" containing the value "1". To its right is a button labeled "Hide Details". Below this, the "Permissions:" section contains four checkboxes: "Readable by Admins" (checked), "Readable by Everyone" (checked), "Writable by Admins" (checked), and "Writable by Everyone" (unchecked). The "TTL Type:" section features a dropdown menu currently set to "Relative". Below that, the "TTL (seconds):" text box contains the value "86400". The "Value Type:" section has a dropdown menu set to "URL". A large text area at the bottom contains the URL "http://example.com/". At the very bottom of the dialog are two buttons: "Cancel" and "Done".

Figure 3.2.15: Show Details Window for "Add Handle Value"

In addition to the settings shown in Figure 3.2.15, the **Show Details** window for an Administrator Value will display all of the administrator permissions possible for that admin value (see Figure 3.2.16). These should be edited with care and only when necessary.

Figure 3.2.16: Add Admin Value

3.2.6.4 Value Types

In addition to URL, EMAIL, and HS_ADMIN, there are additional **Value Types**. They include:

HS_ALIAS – used to add a handle alias as a handle value.

HS_SITE – used to add the site information for prefix handles to indicate where handles with that prefix are resolved.

HS_VLIST – used to define administrator groups with a list of other handle values.

HS_SECKEY – used to add a secret key as a handle value. Typically, the 'public read permission' for this type of value would be turned off.

HS_PUBKEY – used to add a public key as a handle value.

HS_SERV – used to add site information as a handle value.

URN – used to store a value identified as a URN (type not defined or registered).

INET_HOST – used to identify a handle value which has an IP address or host name.

DESC – UTF8-encoded text descriptions of the object identified by the handle

Additional information on Value Types can be found in the [Handle System Informational RFCs](#).

Values can be changed after they have been added by clicking on a value to select it, and then clicking the **Edit** button. Values can be removed before the handle is created by selecting them and clicking **Remove**. A warning message will display to verify that removing the value is the action you intended.

3.2.6.5 Additional Options for Managing New Handle Values

Values can be changed after they have been added by clicking on a value to select it, and then clicking the **Edit** button. Values can be removed before the handle is created by selecting them and clicking **Remove**. A warning message will display to verify that removing the value is the action you intended.

3.2.6.6 Create Handle

When all values have been added and are correct, click **Create Handle**. If the creation is successful, the message displayed below in Figure 3.2.1.7 will display. If there was an error that prevented the handle from being created, the message will display the error instead.

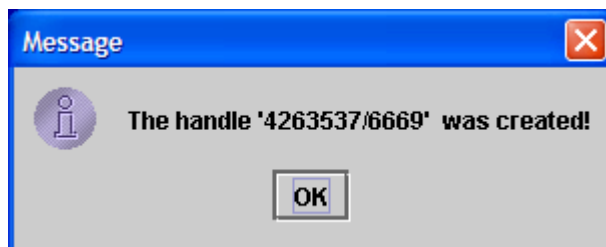


Figure 3.2.17: Successful Creation Message

3.2.7 Modify

To modify a handle means to change one or more of its values, or remove values. (A handle string itself cannot be "modified". If you create handle 123/456 when you meant to create 123/457, you must delete 123/456 and create 123/457.) The terms "modify" and "edit" are frequently used interchangeably.

In the Default Window text box, type the handle you wish to modify, and click **Modify**. If you have not authenticated yourself, the **Handle Authentication** window will display.

Modify a Handle By Changing Its Values

The **Edit Handle** window displays all the existing values that you have permission to edit. To change a value, click on the value to highlight it, and click **Edit**, or highlight the value and double click.

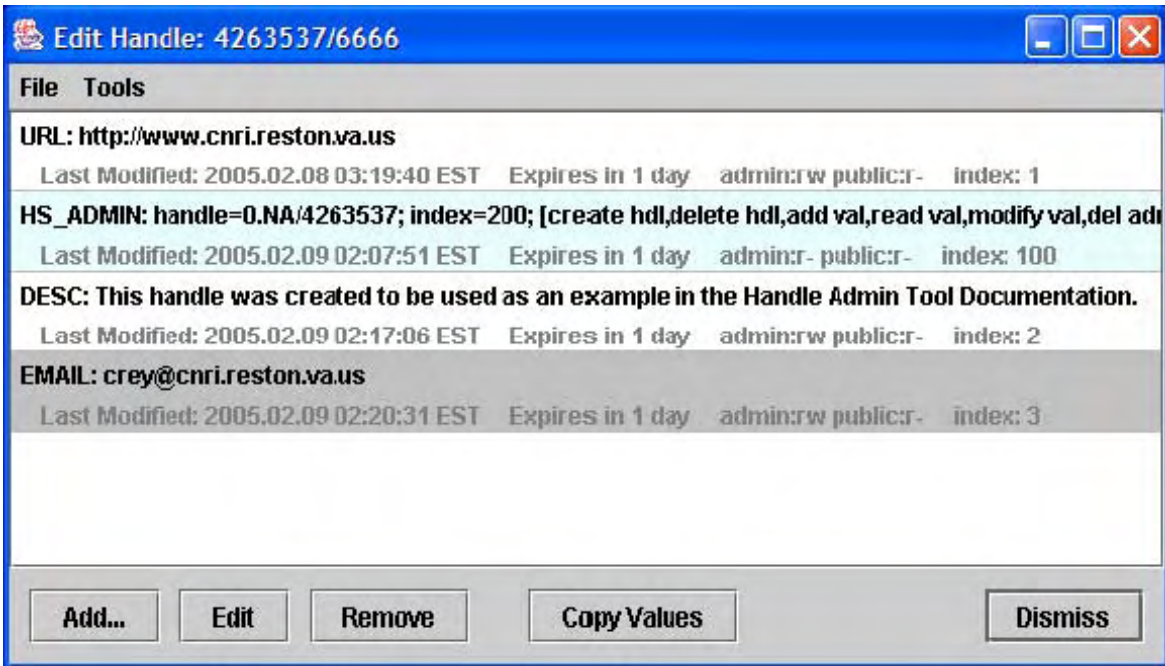


Figure 3.2.18: Edit Handle

The **Edit Handle Value** window will open, displaying the current data with details hidden. The **Index** value will be grayed out. It cannot be changed. If the **Permissions** or the **TTL** value must be changed, click **Show Detail**.

Enter the change and click **Done**.

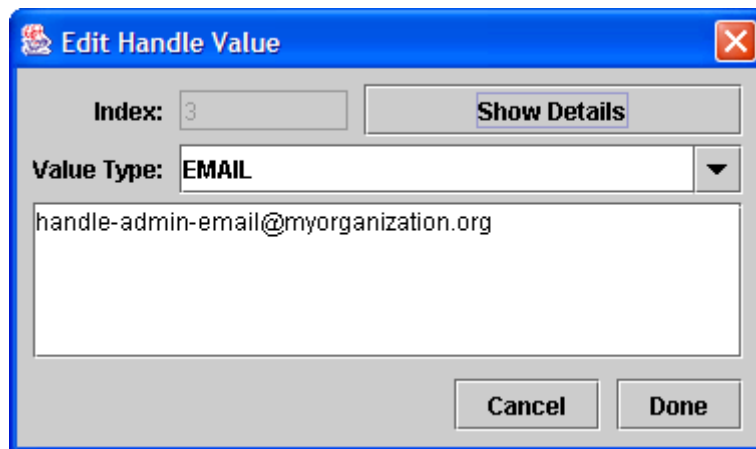


Figure 3.2.19: Edit Handle Value for Email

The Edit Handle window may also be retrieved following a LookUp by clicking on the **Edit Handle** button in the **View Handle** window.

3.2.8 Remove Handle Value

To delete a selected value from a handle (not a handle itself) select and highlight the unwanted value and click the **Remove** button in the **Edit Handle** window. A Delete Confirmation message will appear to confirm your choice before the handle value is removed. (see Figure 3.2.20 below).

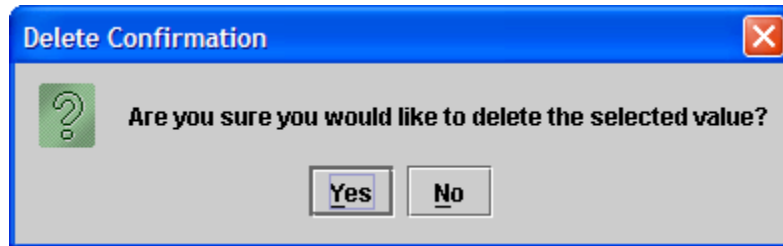


Figure 3.2.20: Remove Handle Value Confirmation

3.2.9 Delete Handle

To delete a handle means to remove it permanently from the Handle System. Enter the handle to be deleted into the Default Window, and click **Delete**. If you have not authenticated yourself, the Handle Authentication window will display.

To remove a value from the handle record, but keep the handle, select the value and click the **Remove** button in the **Edit Handle** window (see Section 3.2.8, **Remove Handle Value**).

A confirmation message will display to verify that deleting the handle is the action you intended. The action cannot be undone.

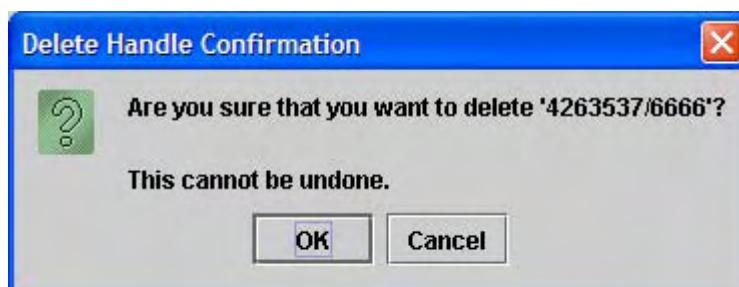


Figure 3.2.21 Delete Handle Confirmation

3.2.10 Batch Processing

Handles can be created, modified or deleted in "batches". All batch files are plain text format lists of create, modify or delete operations for specified handles. One batch file can have more than one type of handle operation. (See Section 3.2.10.2, **Handle Operations** for a list of operations.)

The **Batch Processor** tool in the Handle Tool submits batches to the Handle System, provides output for review including actions taken during the process, and reports success or errors encountered during the process.

3.2.10.1 Sample Batch

A sample batch is illustrated below.

```
CREATE TEST/ts1
  100 HS_ADMIN 86400 1110 ADMIN 300:111111111111:TEST/ts1
  300 HS_SECKEY 86400 1100 UTF8 my_password
  1 URL 86400 1110 UTF8 http://www.handle.net

CREATE TEST.ts2
  100 HS_ADMIN 86400 1110 ADMIN 300:111111111111:0.NA/TEST.ts1
  300 HS_PUBKEY 86400 1110 FILE c:\somewhere\pubkey.bin
  101 HS_ADMIN 86400 1110 ADMIN 301:111111111111:0.NA/TEST.ts1
  301 HS_SECKEY 86400 1110 FILE my_password
  3 URL 86400 1110 UTF8 http://www.cnn.com
  4 EMAIL 86400 1110 UTF8 hdladmin@cnri.reston.va.us
```

3.2.10.2 Handle Operations

The following are the defined handle operations:

(1) Create Handle Batch Format

Operation name is 'CREATE'.

The first line is composed of the following:

CREATE + space + handle_name

The next lines are handle value lines. There must be a handle value line to define the administrator of the handle.

End the CREATE handle operation with a blank line.

See Section 3.2.6.4, **Value Types** for a list of pre-defined handle value types. Each handle value line must start with a unique index number, followed by the handle value type, TTL (the time to live in seconds), the Permission set (admin_read, admin_write, public_read, public_write), and the value data. See the 'Handle Value Line Format' section below for more detail.

Example:

```
CREATE TEST/ts1
  100 HS_ADMIN 86400 1110 ADMIN 300:111111111111:TEST/ts1
  300 HS_SECKEY 86400 1100 UTF8 my_password
  1 URL 86400 1110 UTF8 http://www.handle.net
```

(2) Delete Handle Batch Format

Operation name is 'DELETE'. This operation deletes an existing handle completely.

Every line should resemble the following:

DELETE + space + handle_name

Example:

```
DELETE TEST/ts1
DELETE TEST.ts2
```

(3) Add Handle Value Batch Format

Operation name is 'ADD'. This operation adds new handle values to an existing handle.

The first line is composed of the following:

ADD + space + handle_name

The next lines are handle value lines. There must be a handle value line to define the administrator of the handle.

End the CREATE handle operation with a blank line.

See Section 3.2.6.4, **Value Types** for a list of pre-defined handle value types. Each handle value line must start with a unique index number, followed by the handle value type, TTL (the time to live in seconds), the Permission set (admin_read, admin_write, public_read, public_write), and the value data. See the 'Handle Value Line Format' section below for more detail.

Example:

```
ADD TEST/ts1
5 URL 86400 1110 UTF8 http://www.handle.net/admin.html
6 EMAIL 86400 1110 UTF8 hdladmin@cnri.reston.va.us

ADD TEST/ts
6 URL 86400 1110 UTF8 http://www.cnn.com/entertainment.html
7 URL 86400 1110 UTF8 http://www.cnn.com/show.html
8 EMAIL 8600 1110 UTF8 hdladmin@cnri.reston.va.us
```

(4) Remove Handle Value Batch Format

Operation name is 'REMOVE'. This operation removes one or more handle values from an existing handle.

Every line should resemble the following:

REMOVE + space + indexes:handle_name

Each index is separated by ','.

Example:

```
REMOVE 5:TEST/ts1
REMOVE 5,6,7:TEST/ts5
```

(5) Modify Handle Value Batch Format

Operation name is 'MODIFY'. This operation changes one or more handle values for an existing handle.

The first line is composed of the following:

MODIFY + space + handle_name

The next lines are handle value lines. There must be a handle value line to define the administrator of the handle.

End the MODIFY handle operation with a blank line.

See Section 3.2.6.4, **Value Types** for a list of pre-defined handle value types. Each handle value line must start with a unique index number, followed by the handle value type, TTL (the time to live in seconds), the Permission set (admin_read, admin_write, public_read, public_write), and the value data. See the 'Handle Value Line Format' section below for more detail.

Example:

```
MODIFY TEST/ts1
2 URL 86400 1110 UTF8 http://www.handle.net/newadmin.html
3 EMAIL 86400 1110 UTF8 hdladmin@cnri.reston.va.us

MODIFY TEST/ts2
2 URL 86400 1110 UTF8 http://www.cnn.com/entertainment.html
3 URL 86400 1100 UTF8 http://www.cnn.com/newshow.html
```

3.2.11 Handle Value Line Format

Each handle value line is composed of : value_index + space + value_type + space + ttl + space + permission_set + space + value_data. The value_index is a unique integer within the specific handle. The value_types are: HS_ADMIN, HS_SECKEY, EMAIL, URL, HS_PUBKEY, URN, HS_SERV, HS_VLIST, HS_ALIAS, INET_HOST. ttl: handle's time to live in cache counted by seconds. Default is 86400(24 hours). Permission_set: permission values indicated by 4 characters, '1' is true, '0' is false, order is: admin_read, admin_write, public_read, public_write.

3.2.11.1 About Value Data

If the handle value data defines an Administrator, its data format is:

ADMIN + space + admin_index:admin_permission_set + admin_handle

The admin permission set is twelve characters with the following order: add_handle, delete_handle, add_naming_authority, delete_naming_authority, modify_values, remove_

values, add_values, read_values, modify_administrator, remove_administrator, add_administrator and list_handles.

If the handle value type is one of HS_SECKEY, HS_SERV, HS_ALIAS, EMAIL, URL, URN, and INET_HOST its data will be a string.

The value_data format is: UTF8 + space + string_content

If the handle value data is a local file, its data format is:

FILE + space + file_path

If the handle value data is a value reference list, its data format is:

LIST + space + index1:handle1;index2:handle2;

3.2.11.2 Examples

(1) Handle data in an administration value:

```
100 HS_ADMIN 86400 1110 ADMIN 300:110011111111:0.NA/TEST.ts1
```

Explanation:

100 is index;

HS_ADMIN is type;

86400 is the time to live in cache in seconds;

1110 is the value permissions which allow admin write, admin read, public read;

ADMIN indicates that this value data is an administrator record;

300 is the administrator handle index;

110011111111 defines the administration permissions (add_handle, delete_handle, no add_naming_authority, no delete_naming_authority, modify_values, remove_values, add_values, read_values, modify_administrator, remove_administrator, add_administrator, list_handles);

0.NA/TEST is the administrator handle name;

(2) Handle value data is string:

```
2 URL 86400 1110 UTF8 http://www.handle.net/
```

(3) Handle value data comes from a local file:

```
300 HS_PUBKEY 86400 1110 FILE c:\somewhere\pubkey.bin
2 HS_SITE 86400 1110 FILE c:\somewhere\siteinfo.bin
```

(3) Handle value data is handle value reference list:

```
1 HS_VLIST 86400 1110 LIST 300:100.ADMIN/USER1; 300:100.ADMIN/USER2;
```

(4) Example of all the handle value types:

```
100 HS_ADMIN 86400 1110 ADMIN 300:111111111111:0.NA/TEST
1 HS_SITE 86400 1110 FILE c:\somewhere\siteinfo.bin
2 HS_SERV 86400 1110 UTF8 0.SERV/TEST
300 HS_PUBKEY 86400 1110 FILE c:\somewhere\publickey.bin
301 HS_SECKEY 86400 1100 UTF8 my password
400 HS_VLIST 86400 1110 LIST 300:10.ADMIN/USER1; 300:10.ADMIN/USER2;
7 EMAIL 86400 1110 UTF8 hdladmin@cnri.reston.va.us
8 URL 86400 1110 UTF8 http://www.handle.net
9 URN 86400 1110 UTF8 100/Repository
10 INET_HOST 86400 1110 UTF8 123.45.678.9
```

3.2.12 Using the Batch Processor Tool

You can watch the batch upload process by selecting **Open Batch File** from the File Menu on the Default Window.

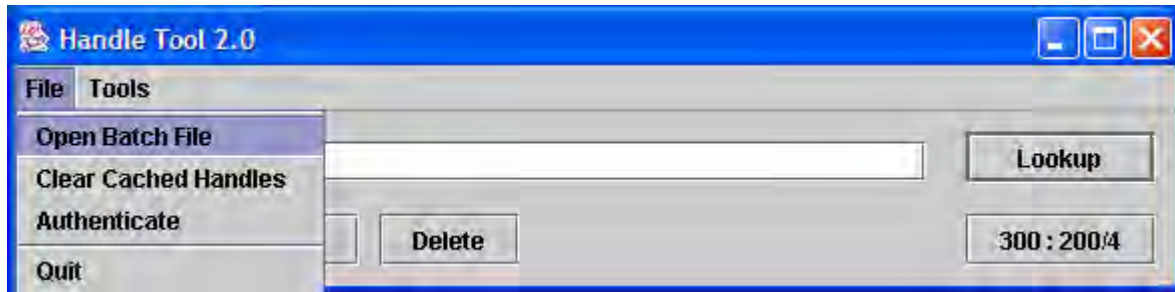


Figure 3.2.22: Open Batch File Option

An Open Batch File window will display with which you can browse the desktop to find the batch file. The Batch Processor will start, and the path to the selected file will display in the Batch Processor window.

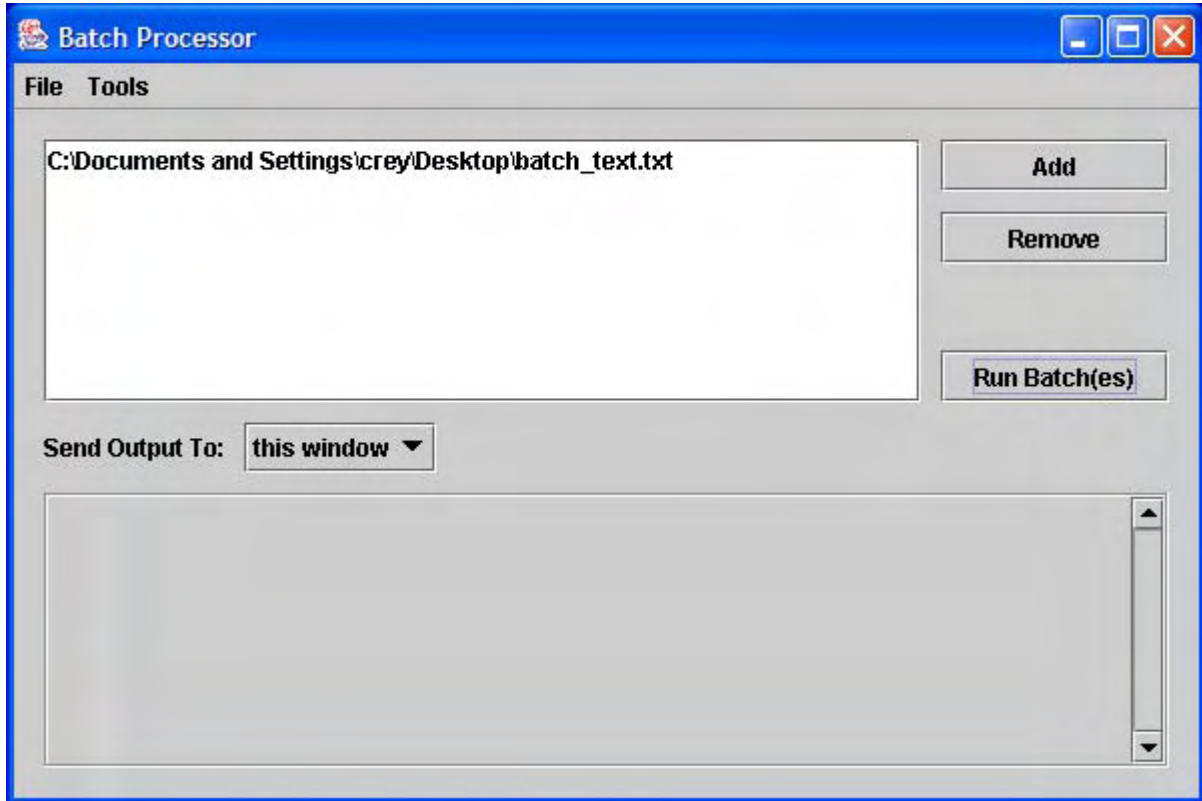


Figure 3.2.23: Batch Processor With Single Batch File Added

The **Batch Processor** can also be started by selecting **Batch Processor** under the **Tools** menu on the Default Window.

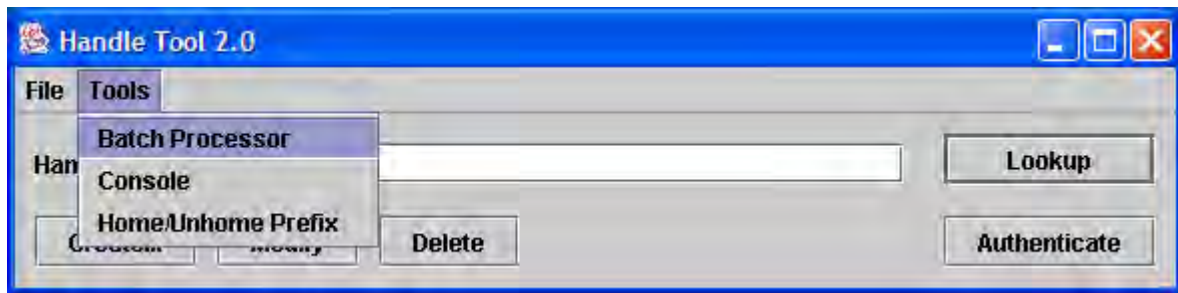


Figure 3.2.24: Starting the Batch Processor

Use the **Add** button to add more batches to the operation, or highlight a batch file name and select **Remove** to delete a previously selected batch file.

The output (log) from the batch process can be directed to display in the Batch Processor window (See Figure 3.2.25) or directed to a file.

If Send Output To: is changed to a file, use the **Choose File** button to bring up the Select Output File window for choosing a file for the output. See Figure 3.2.25 below.

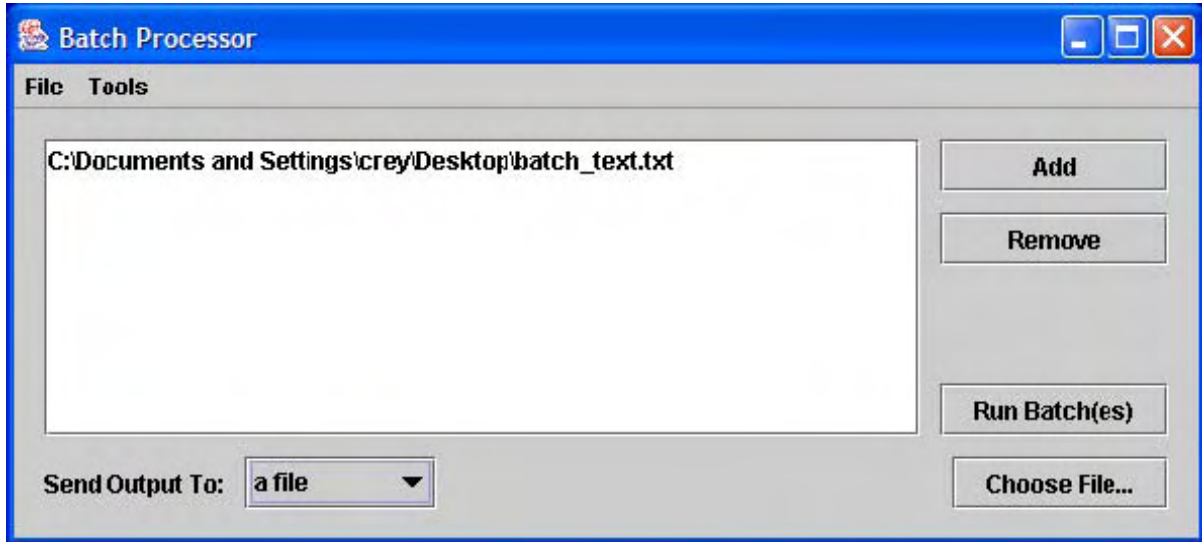


Figure 3.2.25: Send Output To a File

Click on **Run Batch(es)** when all batches have been listed.

If you have not authenticated yourself to the Handle System, the Handle Authentication window will display before the process begins. However, if authentication is specified in the batch file, then that is used instead.

A message will display when the processing is complete.

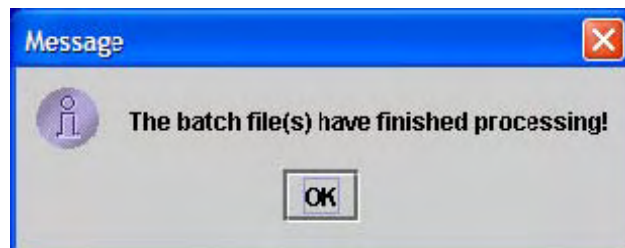


Figure 3.2.26: Processing Complete Message

The Output from a batch process shows the handle operations that were processed, any errors that occurred, and the summary of each batch (number of items, number of lines in the file, number of successful items, and processing time).

The output displayed in Figure 3.2.27 indicates that the batch failed to process, and provides diagnostics explaining what caused the failure. Success/Total Entries would be 1/1 if the batch had processed successfully. The batch should be edited and resubmitted. There would be output for each batch if more than one is submitted.

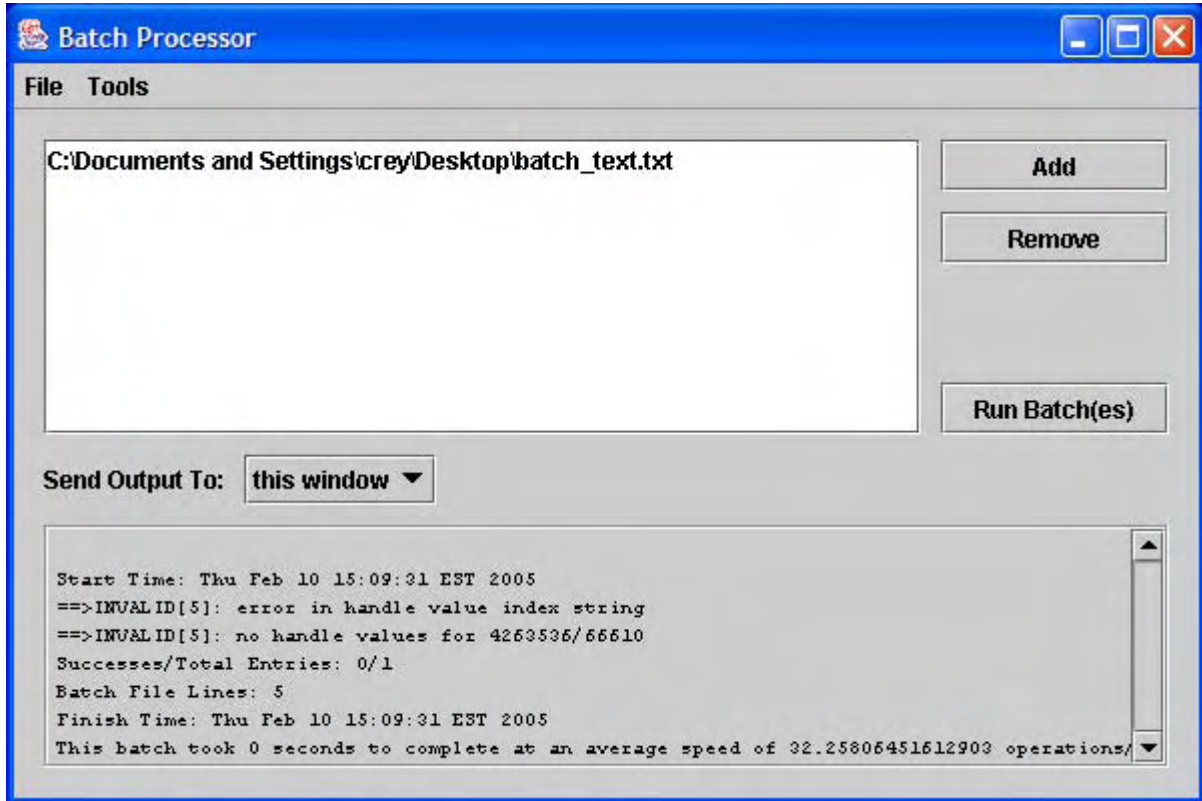


Figure 3.2.27: Batch Processor with Output Displayed

3.2.13 Homing/Unhoming a Prefix

"Homing" a prefix on a particular site tells the server(s) that make up the site that they are responsible for the given prefix. This way, when a resolver comes along and asks for a handle under that prefix, the server can say "Here it is." or "It doesn't exist." or even "Why are you asking me? I don't have it." Permission to Home and Unhome is in the server admins parameter in the config.dct file in the server directory.

From the Tools menu, click on the Home/Unhome Prefix. See Figure 3.2.28 below.

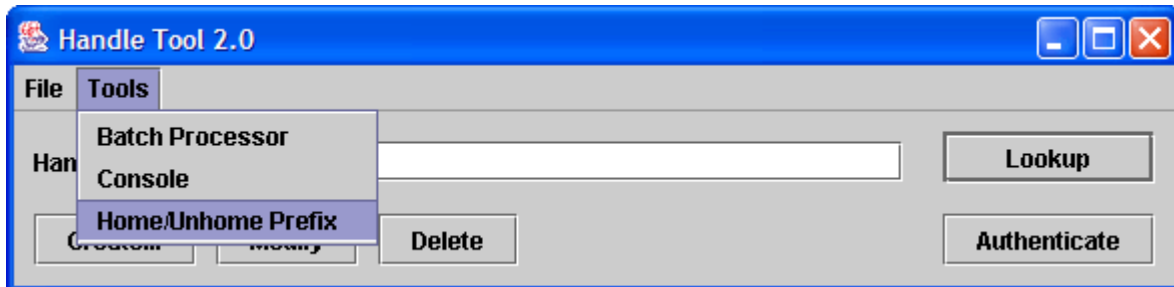


Figure 3.2.28 Home/Unhome Prefix Tool Menu

The window shown in Figure 3.2.28 below will open.



Figure 3.2.29 - Home detail

Enter the prefix to be Homed or Unhomed, and select either "Home Prefix" or "Unhome Prefix". The site information can be provided either by selecting a siteinfo.bin file (default), or by selecting "By Address" and providing the Server Address and Server port number.

4 Batch Operation - Command Line

It may often be desirable to perform more handle operations than it is possible to perform using either of the administration tools. In these cases it is best to use the batch facilities included with the Handle System distribution.

You can submit batches through the Handle Administration Tool, as detailed in Chapter 3, *Using the Handle Administration Tool*. You can also submit batches using the 'GenericBatch' command line utility. It can be invoked using the following command:

```
java -cp handle.jar net.handle.apps.batch.GenericBatch <batchfile>
[<LogFile>]
```

All batch files are plain text format. One batch file can have more than one handle operation. The handle operations are: Create Handle, Delete Handle, Home/Unhome Prefix, Add Handle Value, Remove Handle Value, Modify Handle Value, Authenticate User, Setup Session.

If you need to change authentication for subsequent batch operations, the new authentication information should be put before the batch block. If you authenticate during the batch submission, then you need not include the authentication information in the batch file.

4.1 Create Handle Batch Format

Operation name is 'CREATE'.

The first line is composed of the following:

```
CREATE + space + handle_name
```

The next lines are handle value lines. (See Section 4.9, *Handle Value Line Format*.) There must be a handle value line to define the administrator of the handle. End the 'CREATE' handle operation with a blank line.

The list of predefined handle value types is as follows: HS_ADMIN, HS_VLIST, HS_SECKEY, HS_PUBKEY, HS_SITE, HS_SERV, HS_ALIAS, EMAIL, URL, URN, INET_HOST. Each handle value line must start with a unique index number, followed by the handle value type from the list above, ttl (the time to live in seconds), the permission set (admin read, admin write, public read, public write), and the value data.

See Section 4.9, *Handle Value Line Format* for more detail.

Example:

```
CREATE 12345/hdl1
100 HS_ADMIN 86400 1110 ADMIN
300:11111111111111:12345/hdl1
300 HS_SECKEY 86400 1100 UTF8 my_password
3 URL 86400 1110 UTF8 http://www.handle.net
```

```
CREATE 12345/hdl2
100 HS_ADMIN 86400 1110 ADMIN 200:11111111111111:0.NA/12345
```

```
3 URL 86400 1110 UTF8 http://www.yourorg.org
```

4.2 Delete Handle Batch Format

Operation name is 'DELETE'.

This operation deletes an existing handle completely. Every record is a line with:

```
DELETE + space + handle_name
```

Example:

```
DELETE 12345/hdl1
DELETE 12345/hdl2
```

4.3 (Un)Home Prefix Batch Format

Operation name is 'HOME' or 'UNHOME'.

This operation associates a prefix with a handle server. It only works on existing prefixes and active handle servers. It tells the server which prefixes will be homed or unhomed to it. The first line provides the service information:

```
HOME/UNHOME + space + server_ip:server_port:protocol(tcp,udp,http)
```

The next lines give the prefix names which will be homed/unhomed at this server.

Examples:

```
HOME 10.27.10.28:2641:TCP
0.NA/12345

UNHOME 10.27.10.28:2641:TCP
0.NA/12345
0.NA/TEST1.t1
```

4.4 Add Handle Value Batch Format

Operation name is 'ADD'.

This operation adds new handle values to an existing handle. The first line is composed of the following:

```
ADD + space + handle_name
```

The next lines are handle value lines. (See Section 4.9, *Handle Value Line Format*.) There must be a handle value line to define the administrator of the handle. End the CREATE handle operation with a blank line. The list of predefined handle value types is as follows: HS_ADMIN, HS_VLIST, HS_SECKEY, HS_PUBKEY, HS_SITE, HS_SERV, HS_ALIAS, EMAIL, URL, URN, INET_HOST. Each handle value line must start with a unique index number, followed by the handle value type from the list above, ttl (the time to live in seconds), the permission set (admin read, admin write, public read, public write), and the value data.

```
ADD 12345/hdl1
```

```
5 URL 86400 1110 UTF8 http://www.handle.net/admin.html
6 EMAIL 86400 1110 UTF8 hdladmin@cnri.reston.va.us
```

```
ADD 12345/hdl2
```

```
5 URL 86400 1110 UTF8 http://www.cnri.reston.va.us
6 EMAIL 8600 1110 UTF8 hdladmin@cnri.reston.va.us
```

4.5 Remove Handle Value Batch Format

Operation name is 'REMOVE'.

This operation removes one or more handle values from an existing handle. Every record is a line with:

```
REMOVE + space + indexes:handle_name
```

Each index is separated by ','

Example:

```
REMOVE 5:12345/hdl1
REMOVE 5,6:12345/hdl2
```

4.6 Modify Handle Batch Format

Operation name is 'MODIFY'.

This operation changes one or more handle values for an existing handle. The first line is composed of the following:

```
MODIFY + space + handle_name
```

The next lines are handle value lines. (See Section 4.9, *Handle Value Line Format*.) There must be a handle value line to define the administrator of the handle. End the CREATE handle operation with a blank line. The list of predefined handle value types is as follows: HS_ADMIN, HS_VLIST, HS_SECKEY, HS_PUBKEY, HS_SITE, HS_SERV, HS_ALIAS, EMAIL, URL, URN, INET_HOST. Each handle value line must start with a unique index number, followed by the handle value type from the list above, ttl (the time to live in seconds), the permission set (admin read, admin write, public read, public write), and the value data.

Example:

```
MODIFY 12345/hdl1
2 URL 86400 1110 UTF8 http://www.handle.net/newadmin.html
3 EMAIL 86400 1110 UTF8 hdladmin@cnri.reston.va.us

MODIFY 12345/hdl2
2 URL 86400 1110 UTF8 http://www.cnn.com/newentertainment.html
3 URL 86400 1100 UTF8 http://www.cnn.com/newshow.html
```

4.7 Authentication Information Format

Operation name is 'AUTHENTICATE'.

For secret key authentication:

First line:

```
AUTHENTICATE+space+SECKEY:admin_index:admin_handle
```

Second line:

```
Password
```

Example:

```
AUTHENTICATE SECKEY:301:0.NA/12345  
my_password
```

For private key authentication:

First line:

```
AUTHENTICATE PUBKEY:admin_index:admin_handle
```

Second line:

If your private key was created and encrypted by passphrase, then:

```
private_key_file_path + '|' + passphrase
```

Otherwise:

```
private_key_file_path
```

Example:

```
AUTHENTICATE PUBKEY:300:0.NA/12345  
c:\home\keyfile|my_pass_phrase  
AUTHENTICATE PUBKEY:300:0.NA/12345  
c:\home\keyfile
```

4.8 Session Setup Information Format

Operation name is 'SESSIONSETUP'.

The 'USESESSION' flag is mandatory. Remaining fields are used to specify optional public key pair information, session attributes (e.g., "Encrypted", "Authenticated"), "If session fails, use challenge response" flag and "Timeout".

The first line is composed of the following:

```
SESSIONSETUP
```

Use the following lines to specify mandatory and optional session setup data:

```
USESESSION:<session_on_or_off_flag>  
PUBEXNGKEYFILE:public_exchange_key_file  
PUBEXNGKEYREF:pub_exchange_key_ref_index:pub_exchange_key_  
ref_handle  
PRIVEXNGKEYFILE:private_exchange_key_file  
PASSPHRASE:pass_phrase_for_private_exchange_key  
OPTIONS:<encrypt><authenticate><fallback on challenge  
response>  
TIMEOUT:time_out_in_hours
```

End the 'SESSIONSETUP' operation with a blank line.

In the above lines, the 'USESESSION' flag is mandatory. Either 'PUBEXNGKEYFILE:' or 'PUBEXNGKEYREF:', and 'PRIVEXNGKEYFILE:', 'OPTIONS:', 'TIMEOUT:' are optional. 'PASSPHRASE:' is conditional.

If 'OPTIONS:' is omitted, session messages will neither be encrypted nor authenticated; however, the "If session fails, use challenge response" flag will be set to make sure requests are carried through without session. The 'SESSIONSETUP' line must come first. The remaining lines can be in any order. Do not include a blank line until it ends.

Example 1: Use public exchange key from server.

```
SESSIONSETUP
USESESSION:1
```

Example 2: Use public exchange key from a file (client provides exchange keys).

```
SESSIONSETUP
USESESSION:1
PUBEXNGKEYFILE:c:\hs\bin\PubKey.bin
PRIVEXNGKEYFILE:c:\hs\bin\PrivKey.bin
PASSPHRASE:secret
OPTIONS:111
TIMEOUT:24
```

Example 3: Use public exchange key from a handle value reference (client provides exchange keys).

```
SESSIONSETUP
USESESSION:1
PUBEXNGKEYREF:300:0.NA/12345
PRIVEXNGKEYFILE:c:\hs\bin\PrivKey.bin
```

4.9 Handle Value Line Format

Each handle value line is composed of:

```
value_index + space + value_type + space + ttl + space +
permission_set + space + value_data
```

The value index is a unique integer within the specific handle.

The value types are: HS_ADMIN, HS_SECKEY, EMAIL, URL, HS_PUBKEY, URN, HS_SERV, HS_VLIST, HS_ALIAS.

ttl: handle's time to live in cache counted by seconds. Default is 86400(24 hours).

Permission set: permission values indicated by 4 characters, '1' is true, '0' is false, order is: admin read, admin write, public read, public write.

Value data:

If the handle value data defines an Administrator, its data format is:

```
ADMIN + space + admin index:admin permission set + admin
handle
```


The admin permission set is twelve characters with the following order: add handle, delete handle, add naming authority, delete naming authority, modify values, remove values, add values, read values, modify administrator, remove administrator, add administrator and list handles.

If the handle value type is one of HS_SECKEY, HS_SERV, HS_ALIAS, EMAIL, URL, URN, its data will be a string. The value data format is:

```
UTF8 + space + string_content
```

If the handle value data is a local file, its data format is:

```
FILE + space + file_path
```

If the handle value data is a value reference list, its data format is:

```
LIST + space + index1:handle1;index2:handle2;
```

Examples:

(1) Where handle value data is an administrative record:

```
100 HS_ADMIN 86400 1110 ADMIN
300:110011111111:0.NA/12345
```

Explanation:

100 is index;

HS_ADMIN is type;

86400 is the time to live in cache in seconds;

1110 is the value permissions which allow admin write, admin read, public read;

ADMIN indicates that this value data is an administrator record;

300 is the administrator handle index;

110011111111 defines the administration permissions (add handle, delete handle, no add naming authority, no delete naming authority, modify values, remove values, add values, read values, modify administrator, remove administrator, add administrator, list handles);

0.NA/12345 is the administrator handle name;

(2) Where handle value data is a string:

```
2 URL 86400 1110 UTF8 http://www.handle.net/
```

(3) Where handle value data comes from a local file:

```
300 HS_PUBKEY 86400 1110 FILE
c:\somewhere\pubkey.bin 2 HS_SITE 86400 1110 FILE
c:\somewhere\siteinfo.bin
```

(4) Where handle value data is a handle value reference list:

```
1 HS_VLIST 86400 1110 LIST 300:10.50/USR1; 300:10.50/USR2;
```

(5) Example using some of the registered handle value types:

```
100 HS_ADMIN 86400 1110 ADMIN 300:111111111111:0.NA/12345
1 HS_SITE 86400 1110 FILE c:\somewhere\siteinfo.bin
2 HS_SERV 86400 1110 UTF8 0.NA/12345
300 HS_PUBKEY 86400 1110 FILE c:\somewhere\publickey.bin
301 HS_SECKEY 86400 1100 UTF8 my password
400 HS_VLIST 86400 1110 LIST 300:12346/USR1; 300:12347/USR2;
7 EMAIL 86400 1110 UTF8 hdladmin@cnri.reston.va.us
8 URL 86400 1110 UTF8 http://www.handle.net
9 DESC 86400 1110 UTF8 Info about this handle
```

5 Advanced Server Configuration

A handle server can be further configured through the 'config.dct' file located in its installation directory. This file is divided into various sections, each of which consists of a set of configuration values related to a specific part of the server. These sections are detailed below.

5.1 hdl_http_config

This section contains the configuration variables for the HTTP interface to the HANDLE.NET server. You can access this interface by pointing a web browser at your server's IP and HTTP port.

- `bind_address`: The IP address this interface should use.
- `num_threads`: The number of threads that should be reserved for answering requests.
- `bind_port`: The port which the HTTP interface should use.
- `backlog`: The number of incoming connections that can be queued while all threads are in use.
- `log_accesses`: "yes" or "no". If set to "yes", each access on the HTTP interface will be logged.
- `track_threads`: "yes" or "no". If set to "yes", debugging messages concerning thread usage on the HTTP interface will be printed to stdout periodically.
- `query_page`: Full path and name of the HTML file that your HTTP proxy should use as the query page. If no page is specified, the default Handle System query page will be used.
- `response_page`: Full path and name of the HTML file that your HTTP proxy should use as the data response page. If no page is specified, the default Handle System response page will be used.
- `error_page`: Full path and name of the HTML file that your HTTP proxy should use as error page. If no page is specified, the default Handle System error page will be used.
- `virtual_hosts`: One or more virtual host names and corresponding pages can be specified in this entry. Details of virtual hosts are specified by subentries inside.

An example of virtual host setting would be:

```
"virtual_hosts"=
{  "hostname"="significant.myvirtualhost.com"
  "query_page" ="/home/www/query_page.html "
  "response_page"="/home/www/response_page.html "
  "error_page"="/home/www/error_page.html "
}
```

5.2 hdl_tcp_config

This section contains the configuration variables for the TCP interface to the handle server. The TCP interface is required for handle administration.

- `bind_address`: The IP address the TCP interface should use.
- `num_threads`: The number of threads that should be reserved for answering requests.
- `bind_port`: The port the TCP interface should use.
- `backlog`: The number of incoming connections that can be queued while all threads are in use.
- `log_accesses`: "yes" or "no". If set to "yes", each access on the TCP interface will be logged.
- `track_threads`: "yes" or "no". If set to "yes", debugging messages concerning thread usage on the TCP interface will be printed to stdout periodically.

5.3 hdl_udp_config

This section contains the configuration variables for the UDP interface to the handle server.

- `bind_address`: The IP address the UDP interface should use.
- `num_threads`: The number of threads that should be reserved for answering requests.
- `bind_port`: The port the UDP interface should use.
- `backlog`: The number of incoming connections that can be queued while all threads are in use.
- `log_accesses`: "yes" or "no". If set to "yes", all accesses on the UDP interface will be logged.
- `track_threads`: "yes" or "no". If set to "yes", debugging messages concerning thread usage on the UDP interface will be printed to stdout periodically.

5.4 server_config

This section contains the configuration variables for the server.

- `server_admins`: A list of administrators with permission to un/home prefixes, perform replication, and perform database backups.
- `backup_admins`: A list of administrators with permission to checkpoint the database.

- `replication_admins`: A list of administrators with permission to replicate handles on the server.
- `replication_interval`: Time interval in milliseconds between mirror server updates.
- `replication_authentication`: On mirror handle servers, this setting indicates the authentication handle that the primary server will use when performing handle replication. Should be of the form `handle:index`.
- `this_server_id`: The identification number of this particular server. Used to distinguish from other servers within the same site.
- `max_auth_time`: The number of seconds to wait for a client to respond to an authentication challenge.
- `case_sensitive`: "yes" or "no". Whether or not handles are case sensitive. It is highly recommended to always leave this set to "no".
- `allow_recursion`: "yes" or "no". If set to "yes", the server will act as a proxy for resolving external handles. When a client requests a handle that the server is not responsible for, the server will resolve the handle and return the results, just as if it were stored locally. If `allow_recursion` is set to "no", the proxy will only allow resolution of handles stored on the local handle server. It is usually safe to set this to "yes", which is the default value.
- `preferred_global`: A handle server will often need to resolve prefixes from one of the Global Handle Registry servers. Setting this configuration variable to the IP address of a particular global handle server will force the handle server to always use that global handle server or another server within its site (See Section 1.4, *Design & Storage*). This may be desirable to reduce latencies if there is a global handle server that is significantly closer geographically than the others.
- `max_session_time`: Time in milliseconds that an authenticated client session can persist. Minimum is one minute. See the description of sessions in Section 4.8, *Session Setup Information Format* for more information. Defaults to 24 hours.
- `replication_timeout`: Time in milliseconds before an unresponsive replication operation will timeout. Defaults to 5 minutes.
- `storage_type`: "jdb", "sql", or "custom". This allows manual selection of the storage mechanism used by the server. The "jdb" storage option is a custom hash style database that comes included with the Handle System Server distribution. This is the default storage type. The "sql" option allows use of a SQL database for handle storage. See Chapter 12 *Using a SQL Database for Storage*, for more information. Finally, the "custom" setting directs the handle server to use an external Java™ class specified using the `storage_class` setting.
- `storage_class`: The name of the Java™ class that should be used when `storage_type` is set to custom. This class must implement the `net.handle.hdllib.HandleStorage` interface included with the distribution. It must also be in the Java™ classpath when the handle server is started.

- `server_admin_full_access`: "yes" or "no". If set to "no" the "server_admins" will have default permissions at the prefix level. These include the ability to home and unhome prefixes, perform replication, and perform database backups. If set to "yes" the "server_admins" will have the default permissions to do any prefix level operations as well as handle level operations, such as creating, deleting, and modifying handles. When `server_admin_full_access` is enabled, `server_admins` will be able to modify and delete existing handles, even if they are not explicitly given permission in the handle.
- `allow_list_hdls`: "yes" or "no". If set to "no" the 'list handles' operation will be disabled on the server.

5.5 log_save_config

The code for rotating logs is in the `net.handle.server.Main` class and the `ServerLog` class.

The `ServerLog` looks for an entry named `log_save_config` to determine the rotation method. The value of the `log_save_config` is a hashtable similar to the following:

```
log_save_config = {
    log_save_interval = "Weekly"
    log_save_weekday = "Wednesday"
    log_save_directory = "/var/log/hdl/"
}
```

The `log_save_interval` can be `Monthly`, `Weekly`, `Daily`, or `Never` (the default value is "Never", so that upgrading users won't see an unexpected change). If it is "Weekly" then there is a `log_save_weekday` entry that should contain one of `Sunday` | `Monday` | `Tuesday` | `Wednesday` | `Thursday` | `Friday` | `Saturday`, with the default set to `Sunday`. (English language weekday names are required.)

There is also a `log_save_directory` with the value a directory/path where the log files are to be saved. The default is to store the log files in the server directory. If this is a relative path, it will be interpreted as relative to the server directory.

Note that the `SimpleSetup` tool asks for the log rotation interval and sets up the `config.dct` file properly, but omits the `log_save_directory` and the `Weekly` rotation option, leaving those as manual settings that can be made by those who wish to edit the `config.dct` file by hand.

5.6 Other Settings

These settings do not belong in any particular section.

- `server_type`: This is a single setting. This can be either "server" or "cache". A cache server does not store any handles or answer for a particular prefix, it just acts as a caching gateway. The benefit of clients using a caching server is that they can all make use of the single cache that the caching server provides. A regular HANDLE.NET server will behave just like a caching server if it does not have any prefixes homed to it.

- **Interfaces:** This is a list of interfaces that the server should answer on. It should contain one or more of "hdl_tcp", "hdl_udp", and "hdl_http". If you wish to disable access via a protocol, remove that protocol from this list.
- **trace_resolution:** Setting this to "yes" will cause the handle server to print out debugging messages concerning handle resolution.
- **tcp_timeout:** This is the number of milliseconds that will pass before an outgoing TCP connection fails. This number can be set lower to avoid wasting threads due to broken connections. However, setting too low will cause slow connections to fail unnecessarily.
- **no_udp_resolution:** Setting this to "yes" will prevent the server from resolving external handles using the UDP based handle protocol. This may be necessary to run a handle server from behind a firewall.

5.7 Example config.dct File

The following is an example of a config.dct file:

```
{
"hdl_http_config" = {
"bind_address" = "132.151.1.132"
"num_threads" = "15"
"bind_port" = "8000"
"backlog" = "5"
"log_accesses" = "yes"
}
"hdl_tcp_config" = {
"bind_address" = "132.151.1.132"
"num_threads" = "15"
"bind_port" = "2641"
"backlog" = "5"
"log_accesses" = "yes"
}
"hdl_udp_config" = {
"bind_address" = "132.151.1.132"
"num_threads" = "15"
"bind_port" = "2641"
"log_accesses" = "yes"
}
"server_config" = {
"server_admins" = (
"300:0.NA/1234"
)
"replication_admins" = (
"300:0.NA/1234"
)
"max_session_time" = "86400000"
"this_server_id" = "1"
"max_auth_time" = "60000"
"backup_admins" = (
"300:0.NA/1234"
)
}
```

```
"case_sensitive" = "no"
}
"log_save_config" = {
  "log_save_weekday" = "Sunday"
  "log_save_time" = "00:00:00"
  "log_save_directory" = "/dspace/handle-server"
  "log_save_interval" = "Weekly"
}
"no_udp_resolution" = "y"
"interfaces" = (
  "hdl_udp"
  "hdl_tcp"
  "hdl_http"
)
"server_type" = "server"
}
```


6 Other Tools and Features

The Handle System Server distribution also includes other small utilities for maintaining a handle server. A selection of them are described below.

6.1 DBTool

DBTool is a graphical utility for operating directly on a handle server's built-in database. DBTool can be invoked using the command:

```
java -cp handle.jar net.handle.apps.db_tool.DBTool <serverdir>
```

where <serverdir> is the directory containing the server configuration and database files.

WARNING: Do not run this command on a database that is currently in use by a handle server as it could lead to database corruption.

6.2 DBList/DBRemove

If you would like to operate directly on a handle server's database, but without a GUI, there are two other utilities: DBList and DBRemove.

WARNING: Do not run these commands on a database that is currently in use by a handle server as it could lead to database corruption.

DBList will list all the handles in a handle server database. It can be invoked using the command:

```
java -cp handle.jar net.handle.apps.db_tool.DBList <server dir>
```

DBRemove will remove a specified handle from the database. It can be run using the command:

```
java -cp handle.jar net.handle.apps.db_tool.DBRemove <serverdir>
<handle>
```

6.3 Query Resolver

The handle server distribution includes a simple resolver GUI that may be preferable over the resolution facilities in the Admin Tool. It can be run using the command:

```
java -cp handle.jar net.handle.apps.gui.resolver.Main
```

There is also a command line resolver that can be run using the command:

```
java -cp handle.jar net.handle.apps.simple.HDLTrace <handle>
```

6.4 Test Tool

This tool performs client and local/global handle server diagnostic tests. It is run from the command line and requires certain arguments. The following commands should be run in the same directory as the 'handle.jar' file.

Client Test with no arguments:

Sends a request to each Global server and tests each interface. Also pings each server within the site and reports average round trip time and percent packet loss.

```
java -cp handle.jar net.handle.apps.test.Test client
```

Client Test with prefix argument:

Sends a request to server based on a specified prefix and tests each interface. Also pings each server and reports average round trip time and percent packet loss.

```
java -cp handle.jar net.handle.apps.test.Test client  
<Prefix>
```

Example:

```
java -cp handle.jar net.handle.apps.test.Test client  
0.NA/12345
```

Server Test:

Tests local server by sending a request to the server and testing each interface.

```
java -cp handle.jar net.handle.apps.test.Test server  
<config.dct>
```

Example:

```
java -cp handle.jar net.handle.apps.test.Test server  
\hs\svr_1\config.dct
```

Write Test:

Tests handle operations: add, add value, modify value, delete value, delete. If 'admpriv.bin' is not located in the same directory as 'config.dct', user will be prompted for location.

```
java -cp handle.jar net.handle.apps.test.Test write  
<config.dct>
```

Example:

```
java -cp handle.jar net.handle.apps.test.Test write  
\hs\svr_1
```

Test All:

Performs server test, write test, and Global sites client test.

```
java -cp handle.jar net.handle.apps.test.Test all  
<config.dct>
```

Example:

```
java -cp handle.jar net.handle.apps.test.Test all \hs\svr_1
```

6.5 KeyUtil

The KeyUtil.java program allows decrypting and encrypting of private key files. It can be invoked using the command:

```
java -cp handle.jar net.handle.apps.tools.KeyUtil <privkey.bin>
```

If you chose not to encrypt your key at installation, and later change your mind, use this program to encrypt your existing key, and then send the encrypted file to the Handle System Administrator.

6.6 GetRootInfo

GetRootInfo retrieves a current copy of the Global service information for the Handle System. This is the information found in the 'root_info' file.

Usage:

```
java -cp handle.jar net.handle.apps.tools.GetRootInfo <root-server> <port> <outfile>
```

6.7 GetSiteInfo

GetSiteInfo retrieves the service information for a handle server. This is the information found in the 'siteinfo.bin' file.

Usage:

```
java -cp handle.jar net.handle.apps.tools.GetSiteInfo <server> <port> <outfile>
```

6.8 DoCheckpoint

DoCheckpoint is a command line tool to send a checkpoint/backup command to all of the handle servers in a specified site.

Execution is done with the following command:

```
java -cp handle.jar net.handle.apps.tools.DoCheckpoint [siteinfo] [adminhdl] [adminindex] [keytype] [key]
```

Keytype can be either PRIVATE or SECRET. If PRIVATE, then key is the name of a file where the private key is stored. If SECRET, then key is the secret key itself.

7 Replication

The handle server allows for automatic replication of handles to one or more mirror servers. These mirror servers can be used to provide redundancy for resolution or simply as backup for disaster recovery. In sites with multiple servers, handles are distributed evenly across the site. It should be noted that mirror servers cannot be used for handle administration.

When servers are in the same site, the handles for that service are distributed evenly across the site. Mirrors should be able to replicate from either the primary or existing mirrors.

7.1 Setting up a Single Mirror Handle Server

See Chapter 5, *Advanced Server Configuration*, for details on the configuration settings outlined below.

- Run SimpleSetup as explained in the 'INSTALL.txt' file. Be sure to choose "Mirror Server".
- Modify the mirror's **config.dct** :
 - server_admins, as described in the 'Install.txt' file.
 - this_server_id (if more than 1 server in HS SITE),
 - replication_authentication ("privatekey:index:handle")
- Modify the primary server's configuration:
 - Add the replication public key value ('replpub.bin') generated when you ran SimpleSetup, to the handle specified in replication_authentication in the mirror's config.
 - Add that handle to the primary server's replication_admins group in its **config.dct** file.
- ONLY if mirror servers are under the same HS_SITE, save the prefix's second site ('siteinfo.bin' for both mirrors) into each mirror server's directory.

The 'txnsrscsv.bin' file is the 'siteinfo.bin' file from the primary server. The 'txnstat.dct' file will be created once the server has replication information to store. When the mirror server is first started, the server has to "dump" all of the handles from the primary server. When that process is done, the mirror server creates and saves the 'txnstat.dct' file with the current transaction ID from each primary server.

7.2 Setting up a Second Mirror Handle Server

If the 1st mirror server's db file is relatively large, it may be necessary to copy the 'handles.jdb', 'nas.jdb', 'txnstat.dct' files to the second mirror's directory before starting the second mirror server. Be sure the 1st mirror server is shut down while copying the files.

8 Using Custom Handle Storage

This section explains how to configure your HANDLE.NET server to use a database for handle storage other than the built-in database. Instructions follow for using Berkeley DB JE, SQL, and PostgreSQL.

8.1 Enabling Berkeley DB JE Database Support

The Berkeley DB JE database (referred to here as BDBJE) is a very efficient, scalable and widely used open source database. This section explains how to configure your HANDLE.NET server to use BDBJE instead of the built-in database. The BDBJE support in the handle server was built using BDBJE version 2.0.83.

1. Make sure that the "je.jar" file is in your classpath. Ideally it will be located in the same folder as the handle.jar file. If the je.jar file was not included with your handle software, you can download it from the producer of BDBJE, Sleepycat Software, at <http://sleepycat.com/>.

2. Perform the normal setup process for your server, which produces a server directory that contains a config.dct file.

3. Edit the config.dct file so that the section labeled "server_config" (delimited by curly braces) contains the following line:

```
"storage_type" = "BDBJE"
```

4. If you wish to configure additional settings specific to BDBJE you can add any of the following lines. The values below are the default values for each setting.

```
"db_directory" = "<server folder>"
```

This tells the BDBJE which folder should contain the database files. The default is to use the server directory.

```
"enable_recovery_log" = "false"
```

This tells the storage module if it should or should not keep a log of all changes to the database in a dbtxns.log file in the database directory. These changes are recorded at the handle level. The default is not to record these transactions since BDBJE has its own checkpoint/recovery system.

```
"bdbje_dump_on_checkpoint" = "true"
```

This tells the storage module if it should or should not dump a copy of the entire database when a checkpoint is performed. When a checkpoint is performed, the BDBJE checkpoint operation is invoked, but a backup of the database is only made if this option is set to "true".

```
"bdbje_no_sync_on_write" = "false"
```

This tells BSBJE to write changes to the database, but do not synchronize afterwards. Setting this to true improves performance, with a slight cost in reliability (which may be negated when using a journaling file system).

```
"bdbje_enable_status_handle" = "true"
```

This tells the storage module to send database status information in response to a query for the 0.SITE/DB_STATUS handle, as long as that handle doesn't already exist in the database.

8.2 Using a SQL Database for Storage

Using a SQL database as storage for a handle server allows greater control over data deposits as well as permitting complex data query.

8.2.1 Configuring the Handle Server

To configure a handle server with an SQL storage module, first run the SimpleSetup program for the HANDLE.NET software. Once the setup process is completed, a directory will exist that contains the files necessary to run the new handle server.

In the directory for the new handle server is a file named 'config.dct' that can be modified using a text editor. The 'config.dct' file contains all of the settings for the handle server. The 'config.dct' file has some server-wide settings as well as several subsections that affect different parts of the server.

For example, the 'config.dct' file for most handle servers will have sections named hdl tcp config, hdl udp config and hdl http config. Each of these sections holds the settings for one type of "listener" for the handle server.

Normal handle servers (as opposed to simple handle caching servers or http gateways) will also have a section named server config that maintains the settings for the core part of the server. To tell the server to use an SQL backend for storing and retrieving the handles, add the following value to the server config section:

```
storage_type = "sql"
```

Since the specified storage type for this handle server is SQL, some extra settings need to be provided. The following subsection should also be added to the server config section:

```
sql_settings = {
  sql_url = "jdbc:sybase:Tds:localhost:2638"
  sql_driver = "com.sybase.jdbc.SybDriver"
  sql_login = "sqluser"
  sql_passwd = "sqlpassword"
  sql_read_only = "no"}
```

Of course you will need to change the values to suit your particular installation. Here is an informal description of what each item in this section is for:

- sql url: This setting should specify the JDBC URL that is used to connect to the SQL database. Consult the documentation for the database or JDBC driver for a description of what this setting should look like.
- sql driver: This is the name of a Java™ class that contains the driver for the JDBC connection. Consult the documentation for the database or JDBC driver for a description of what this setting should look like.
- sql login: The user name that should be used by the handle server to connect and perform operations on the database.
- sql passwd: The password that should be used by the handle server to connect and perform operations on the database.
- sql read only: a boolean setting (can be "yes" or "no") that indicates whether or not the server should ever need to modify the database in any way. This is a safeguard used for query-only handle servers.

8.2.2 Example SQL Tables

The default configuration assumes a specific database table setup. The following tables were used for RDBMS storage using [MySQL](#).

```

create table nas (
  na varchar(255) not null,
  PRIMARY KEY(na)
) ;
create table handles (
  handle varchar(255) not null,
  idx int4 not null,
  type blob,
  data blob,
  ttl_type int2,
  ttl int4,
  timestamp int4,
  refs blob,
  admin_read bool,
  admin_write bool,
  pub_read bool,
  pub_write bool,
  PRIMARY KEY(handle, idx)
);

```

These tables were used for Oracle.

```

create table handles (
create table nas (
na raw(512)
) ;
      create table handles (
        handle raw(512),
        idx number(10),
        type raw(128),
        data raw(600),

```

```

        ttl_type number(5),
        ttl number(10),
        timestamp number(10),
        refs varchar2(512),
        admin_read varchar2(5),
        admin_write varchar2(5),

        pub_read varchar2(5),
        pub_write varchar2(5)
    ) ;

```

8.2.3 Depositing Handles Outside the Handle Server

If you wish to create or modify handles in the SQL database using custom tools, rather than the handle server, you must use all capital letters for data in the "handle" field, since the Handle System is case insensitive.

8.2.4 Using Custom SQL Statements

It is also possible to specify the SQL that is used by the handle server to query the database. Changing these SQL statements is required if you do not use the same setup as above.

The SQL handle storage object used by the handle server has default SQL statements that are used to query and update the database. To replace the default SQL statements with custom statements, simply add the corresponding configuration setting to the sql settings section described above. The following is a list of the SQL statements, their configuration setting, default values, and a short description of what the statement is used for.

8.2.4.1 get handle stmt

Default:

```

select idx, type, data, ttl_type, ttl, timestamp, refs,
admin_read, admin_write, pub_read, pub_write from handles
where handle = ?

```

Description:

This statement is used to retrieve the set of handle values associated with a handle from the database.

Input: The name of the handle to be queried.

Output:

idx positive integer value; unique across all values for the handle

type alphanumeric value; indicates the type of the data in each value

data alphanumeric value; the data associated with the value ttl type byte/short;
0=relative, 1=absolute

ttl numeric; cache timeout for this value in seconds (if ttl type is absolute, then this indicates the date/time of expiration in seconds since Jan 1 0:00:00 1970.

timestamp numeric; the date that this value was last modified, in seconds since Jan 1 0:00:00 1970

refs alphanumeric; list of tab delimited index:handle pairs. In each pair, tabs that occur in the handle part are escaped as \t.

admin read boolean; indicates whether clients with administrative privileges have access to retrieve the handle value

admin write boolean; indicates whether clients with administrative privileges have permission to modify the handle value

pub read boolean; indicates whether all clients have permission to retrieve the handle value

pub write boolean; indicates whether all clients have permission to modify the handle value

8.2.4.2 have na stmt

Default:

```
select count(*) from nas where na = ?
```

Description:

This statement is used to query whether or not the specified prefix is "homed" to this server.

Input: The prefix (eg 0.NA/12345)

Output: One row, with one field. The value of that field is >0 if this server is responsible for the given prefix, or <=0 if not.

8.2.4.3 del na stmt

Default:

```
delete from nas where na = ?
```

Description:

This statement is used to remove a prefix from the list of prefixes for which this server is responsible.

Input: The prefix handle (e.g., 0.NA/12345)

Output: None

8.2.4.4 add na stmt

Default:

```
insert into nas ( na ) values ( ? )
```

Description:

This statement is used to add a prefix to the list for which this server is responsible.

Input: The prefix to "home" (e.g., 0.NA/12345)

Output: None

8.2.4.5 scan handles stmt

Default:

```
select distinct handle from handles
```

Description:

This statement is used to get a list of all of the handles in the database.

Input: None

Output: a row for each distinct handle in the database.

8.2.4.6 scan by prefix stmt

Default:

```
select distinct handle from handles where handle like ?
```

Description:

This statement is used to get a list of all handles in the database that have a given prefix.

Input: The prefix, including the slash (/) character

Output: A row for each distinct handle in the database that starts with the given prefix

8.2.4.7 scan nas stmt

Default:

```
select distinct na from nas
```

Description:

This statement is used to get a list of distinct prefixes that call this server home.

Input: None

Output: A row for each distinct prefix

8.2.4.8 delete all handles stmt

Default:

```
delete from handles
```

Description:

This statement is used to delete all of the handles in the database (!) This is only used when the handle server is acting as a secondary/mirror to a primary service and has gotten so far out of sync that it tries to delete and recopy the entire database from the primary.

Input: None

Output: None

8.2.4.9 delete all nas stmt

Default:

```
delete from nas
```

Description:

This statement is used to delete all of the prefixes in the database. This is only invoked under the same circumstances as delete all handles stmt.

Input: None

Output: None

8.2.4.10 create handle stmt

Default:

```
insert into handles ( handle, idx, type, data, ttl_type,
  ttl, timestamp, refs, admin_read, admin_write, pub_read,
  pub_write) values ( ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
```

Description:

This statement is used to insert a single handle value into the database.

Input: The fields of the handle and handle value, with the same order and type specified in the default statement above. See get handle stmt for type information for each field.

Output: None

8.2.4.11 handle exists stmt

Default:

```
select count(*) from handles where handle = ?
```

Description:

This statement is used to query whether or not a given handle exists in the database.

Input: The handle being queried

Output: None

8.2.4.12 delete handle stmt

Default:

```
delete from handles where handle = ?
```

Description:

This statement is used to delete a given handle from the database.

Input: The handle being deleted

Output: None

8.2.4.13 modify value stmt

Default:

```
update handles set type = ?, data = ?, ttl_type = ?, ttl =  
?, timestamp = ?, refs = ?, admin_read = ?, admin_write = ?,  
pub_read = ?, pub_write = ? where handle = ? and idx = ?
```

Description:

This statement is used to update a single handle value with new values. The value to update is identified by the handle and index.

Input:

type - alphanumeric; the new type for the handle value

data - alphanumeric; the new data value

ttl_type - byte/short int; indicates whether the cache timeout is specified in relative or absolute terms (1=absolute, 0=relative)

ttl - numeric; indicates the cache timeout in seconds (if ttl type is absolute then the ttl indicates the expiration date in seconds since Jan 1 0:00:00 1970)

timestamp - numeric; date of the last modification to this handle value (should be the current date/time!)

refs - alphanumeric; tab delimited list of references for this handle value. See get handle stmt for encoding.

admin_read - boolean; indicates whether clients with administrative privileges have access to retrieve the handle value

admin_write - boolean; indicates whether clients with administrative privileges have permission to modify the handle value

pub_read - boolean; indicates whether all clients have permission to retrieve the handle value

pub_write - boolean; indicates whether all clients have permission to modify the handle value

8.3 Using PostgreSQL Database

The following instructions, provided courtesy of handle users at the Max Planck Institute for Psycholinguistics, are for setting up a PostgreSQL database for handle storage.

As postgres:

```
createuser -PEDA handleserver
```

Make sure to define a password for that user. Add to

```
/var/lib/pgsql/data/pg_hba.conf
```

```
host handlesystem handleserver 192.168.0.0/16 md5
```

(This assumes that your intranet uses 192.168.x.x IP addresses.)

Activate the new account:

```
pg_ctl restart -D /var/lib/pgsql/data/
```

(You may, depending on your configuration, have to replace /var/lib/pgsql/data here and above with something else.)

As an alternative to

```
pg_ctl restart
```

you may use:

```
/etc/init.d/postgresql restart
```

Create the database and make sure that it uses Unicode:

```
createdb -O handleserver -E unicode handlesystem
```

Now use the psql shell to create the tables, etc.:

```
psql -h yourservername -U handleserver -d handlesystem
create table nas (na bytea not null, primary key(na));
create table handles (handle bytea not null, idx int4 not
null, type bytea, data bytea, ttl_type int2, ttl int4, timestamp
int4, refs text, admin_read bool, admin_write bool, pub_read bool,
pub_write bool, primary key(handle, idx));
create index dataindex on handles ( data );
create index handleindex on handles ( handle );
grant all on nas,handles to handleserver;
grant select on nas,handles to public;
\q
```

The `\q` leaves `psql`. Note that many columns are bytes, not text.

To backup and restore your handle database, use:

```
to backup: pg_dump handlesystem -F t | gzip -c > handletable.tgz
to list: zcat handletable.tgz | pg_restore -F t -l
to restore: zcat handletable.tgz | pg_restore -F t
```

(With "ddlutils", you can also backup / restore between various databases and XML files, which might be useful for some people.)

To get a description of a database or table, in `psql`, use:

```
\d (describes the whole database)
\d tablename (describes one table)
```

(As usual, use `\q` to leave `psql` again. Note that `psql` also has nice features like history (cursor up/down) and tab completion.)

To "defragment" and auto-tune for the current contents, use in `psql`:

```
vacuum analyze handles;
```

Do this from time to time, especially after larger writes, to gain speed.

The `config.dct` section for a PostgreSQL database:

```
"storage_type" = "sql"
"sql_settings" = {
"sql_url" = "jdbc:postgresql://YourServerIPAddress/handlesystem"
"sql_driver" = "org.postgresql.Driver"
"sql_login" = "handleserver"
"sql_passwd" = "yourpassword"
"sql_read_only" = "no"
}
```

When you start the handle server, you must have the JDBC for your database in your classpath. For example:

```
java -cp handle.jar:postgresql8jdbc3.jar net.handle.server.Main
/hdata/
```

(`hdata` is the directory where you have your `config.dct` and so on.)

For the GUI, you still only need `handle.jar`, as usual:

```
java -cp handle.jar net.handle.apps.gui.hadmin.HandleTool
```

Note: These instructions are included courtesy of handle users at the [Max Planck Institute for Psycholinguistics](#) and [Lund University Libraries NetLab](#). It is possible that your settings may differ slightly from those in the examples above.

9 The Handle HTTP Proxy

Proxy servers are not part of any Handle System administration or authentication hierarchy. The Handle System protocol does not authenticate any response from a proxy server. Use of a proxy server is a client option, and the client may have to rely on the proxy server to authenticate any service response from Handle System service components. Clients are responsible for setting up their own trust relationship with the proxy server they select.

9.1 Using the Proxy

Unless you disabled the http interface after setting up your handle server, handles can be directly resolved on your handle server using a web browser. A handle server used in this manner is often referred to as a proxy server.

Using HTTP URLs allows handles to be resolved from standard web browsers without additional client software, but requires that the handles be associated with a specific proxy server. If that proxy server changes its DNS name or otherwise becomes invalid, the reference (i.e., the HTTP URL) to the handle will break. Thus selection or use of proxy servers should be carefully evaluated.

You can connect to your server's HTTP interface by opening a URL like `http://127.0.0.1:8000`. Replace 127.0.0.1 with the IP address or hostname of your handle server. If you changed the HTTP port for the server, replace 8000 with the correct port number. You should see a page like the one below*.

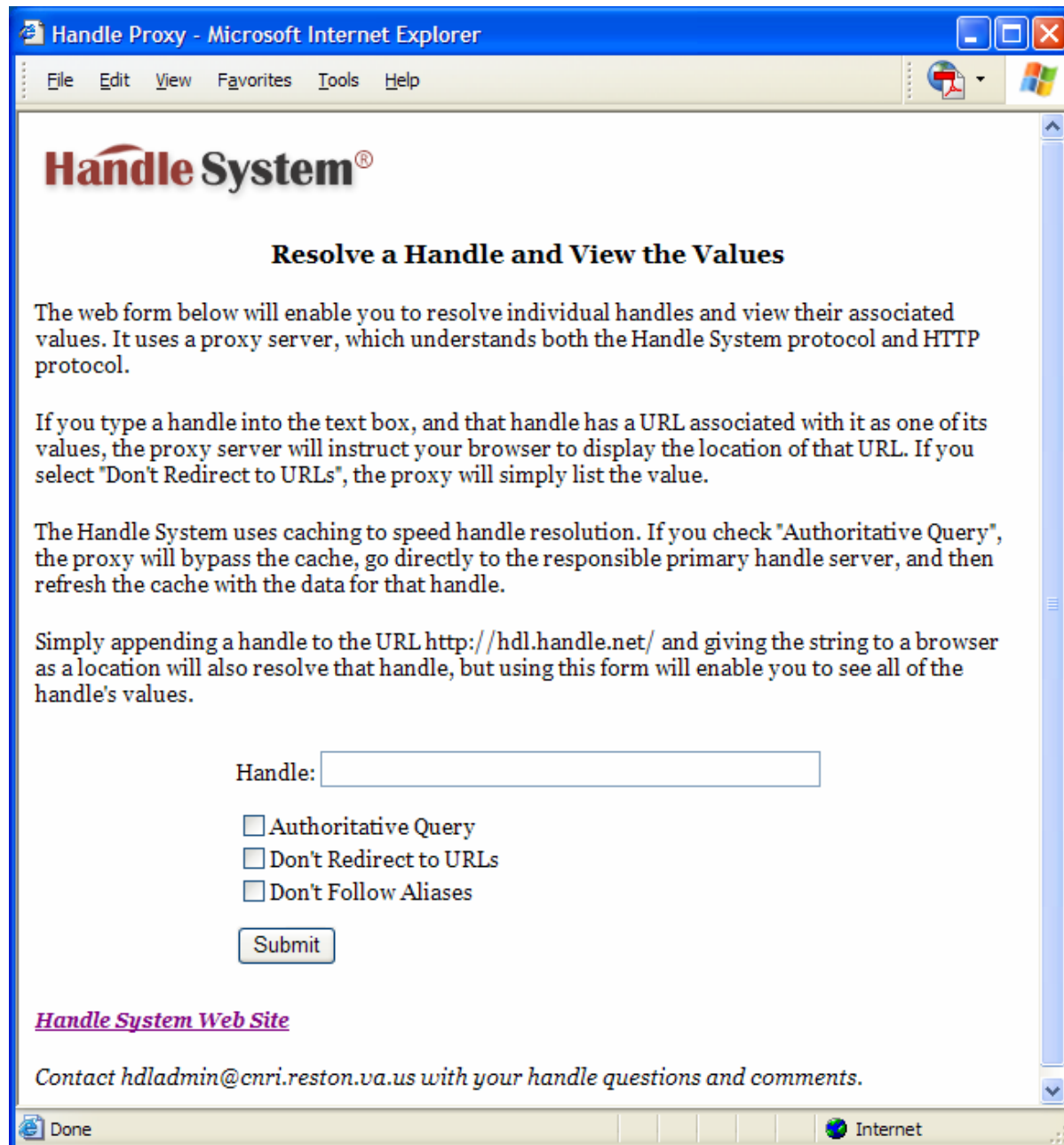


Figure 6.1: Proxy Web Form

****Please note that this web form is not part of the standard HANDLE.NET protocol and may change or be completely removed from future versions of the software.***

It is also possible to build URLs to the proxy which will automatically resolve or redirect to a specified handle. For a handle server with an IP address of 127.0.0.1 and HTTP interface port 8000 the handle 'my_handle00' can be resolved from a web browser through the URL `http://127.0.0.1:8000/my_handle`.

If the 'allow_recursion' option is set in the server's configuration (`config.dct`) file, the HTTP interface will allow resolution of handles which are not stored on the local handle

server. When a client requests an external handle, the handle server will resolve the handle and return the results, just as if it were stored locally. This is how public proxy servers like hdl.handle.net and dx.doi.org are configured. If 'allow_recursion' is disabled, the proxy will only allow resolution of handles stored on the local handle server.

9.2 Using Custom HTML Pages

The HANDLE.NET proxy code supports customization of the query, response and error pages. Simple customization of the pages can be performed by modifying copies of the template files included in the 'handle.jar' file. These templates are located in the jar directory 'net/handle/server/html'.

If new templates are created, the server configuration must be modified to use them. See Section 5.1, *hdl_http_config* for instructions.

9.3 Web Interface for Handle Administration

The HANDLE.NET distribution comes with Java™ servlets for handle administration. These servlets are meant to run in a web server using a Java™ servlet engine like [Apache Tomcat](http://tomcat.apache.org/).

To compile servlets, get the servlet API classes by installing the JSDK (Java™ Servlet Development Kit) and include the JSDK classes in your CLASSPATH. (For information see <http://java.sun.com/products/servlet/>).

These servlet API classes may be included with servlet engines like Apache Tomcat as a jar file (usually "servlet-api.jar"). If you already have a servlet engine, simply add the jar file to your CLASSPATH to compile servlets.

Instructions for use:

1. Install Java™ version 1.4.2 or greater on your computer.
2. You will need to have a web server that supports Java™ servlets.
3. Download, gunzip, and untar the HANDLE.NET distribution.
4. Extract the 'src.jar' file from the handle server directory. The files for setting up the web interface are in 'net/handle/apps/admin_servlets'. The servlet code, for web administration and handle resolution, is four files, and there is one corresponding 'htdocs' directory containing HTML files that will be returned from the servlet/proxy.
5. In 'Admin.java', change the value of the YOUR_NAMING_AUTHORITY variable to your prefix. Change the value of the 'ADMIN_NA' variable to your administrative handle. The default index values for SEC_KEY_IDX is 300 and ADMIN_GROUP_IDX is 200. Include the 'handle.jar' file and the servlet library from your servlet engine (usually 'jsdk.jar') with your CLASSPATH variable. Compile 'Admin.java'.
6. You will need to modify the HTML files under

```
'net/handle/apps/admin_servlets/htdocs'
```

to fit your specific purpose. Change the email address and prefix found in the following html files: `admin_footer.html`, `admin_login.html`, `help.html`, `index.html`, `qform.html`.

7. 'Admin.java' uses two parameters, `TEMPLATE_DIR_KEY` and `BATCH_DIR_KEY`, which are specified in the servlet properties file.

`TEMPLATE_DIR_KEY`: path to html files

`BATCH_DIR_KEY`: path to batch directory

8. In an Apache JServ setup, the `zone.properties` file should be modified as follows:

```
servlet.net.handle.apps.admin_servlet.Admin.initArgs=  
  admin_servlets.html_template_dir=<path to the html files>,  
  admin_servlets.dir=<path to the batch directory>
```

9. In an Apache Jakarta Tomcat setup, the `web.xml` file should be modified as follows:

```
<initparam>  
  <paramname>webadmin_servlets.html_template_dir</paramname>  
  <paramvalue><path to the html files></paramvalue>  
  <paramname>webadmin_servlets.batch_dir</paramname>  
  <paramvalue><path to the batch directory></paramvalue>  
</initparam>
```

10. Add the servlet files to your web server or servlet engine.

11. To access the Admin servlet, use the following syntax:

```
http://host/servlets/net.handle.apps.admin_servlets.Admin
```

You need to add the servlet files to our web server or servlet engine.

See configuration file for mount point for servlet zones.

After setting up and accessing the administration servlet you must login. The administration servlet uses secret key authentication. Create an administrative handle for use in the servlet. For example, if you plan to administer handles under the prefix 0.NA/345678, create the handle 345678/admin to use for web administration. Create the handle with the following values using one of the handle administration tools described in Chapter 3:

- An `HS_ADMIN` value at index 100, with Admin ID Handle=345678/admin, Admin ID Index=300 and any permissions you want.
- An `HS_SECKEY` value at index 300 with any value, e.g., "secretkey". This value will be your login password.

The handle (345678/admin) must be added to the admin group list in the prefix record on GHR.

When you access the administration servlet through your web browser, do the following to log in:

Handle Prefix: 345678

User ID: admin

Password: your_secretkey

10 Handle Clients & the Client Library (Java™ Version)

Communicating with the Handle System is accomplished by sending requests to servers which then return a response. To resolve a handle, a `ResolutionRequest` is sent to a server. To create a handle, a `CreateHandleRequest` is sent. To modify, remove, or add values to (or from) a handle, a `ModifyValueRequest`, `RemoveValueRequest`, or `AddValueRequest` is sent to a server.

There is an object for each of these requests in the `net.handle.hdllib` java package. One way to send these messages to a server is to use a `HandleResolver` object which is located in the `net.handle.hdllib` package. For most messages, the `HandleResolver` object will locate the server that your messages should go to, send them, and return the response that was sent by the server. The following is an example that shows one way of programmatically resolving a handle:

```
import net.handle.hdllib.*;
...
// Get the UTF8 encoding of the desired handle.
byte someHandle[] = Util.encodeString("45678 /1");

// Create a resolution request.
// (without specifying any types, indexes, or authentication info)
ResolutionRequest request =
    new ResolutionRequest(someHandle, null, null, null);

HandleResolver resolver = new HandleResolver();

// Create a resolver that will send the request and return the
// response.
AbstractResponse response = resolver.processRequest(request);

// Check the response to see if the operation was successful.
if(response.responseCode == AbstractMessage.RC_SUCCESS) {

// The resolution was successful, so we'll cast the response
// and get the handle values.
HandleValue values[]
    = ((ResolutionResponse)response).getHandleValues();
for(int i=0; i < values.length; i++) {
    System.out.println(String.valueOf(values[i]));
}
}
```

To simply resolve a handle, the much simpler `resolveHandle` method of the `HandleResolver` can be used, as shown below.

```
import net.handle.hdllib.*;
...
HandleValue values[] =
    new HandleResolver().resolveHandle("12345/1", null, null);
for(int i=0; i < values.length; i++){
    System.out.println(String.valueOf(values[i]));
}
```

}

The HANDLE.NET software distribution include a "simple" package with command line tools to create, delete, and list handles. It also includes programs to home a prefix and trace handle resolution. These programs provide a good starting point and simple guide to developing Java™-based custom handle client software with the API. Each example program includes steps needed to form a handle request to send to a handle server. The programs are run from the command line and require certain arguments. The following commands should be run in the same directory as the 'handle.jar' file.

- **Create Handle:**

Simple tool for handle creation. It uses public key authentication.

```
java -cp handle.jar net.handle.apps.simple.HDLCreate <auth
handle> <auth index> <privkey> <handle>
```

- **Delete Handle:**

Simple tool for handle deletion. It uses public key authentication.

```
java -cp handle.jar net.handle.apps.simple.HDLDelete <auth
handle> <privkey> <filename_of_file_with_handles_to_delete>
```

- **List Handles:**

Simple tool for listing handles. It uses public key authentication.

```
java -cp handle.jar net.handle.apps.simple.HDLList <auth
handle> <auth index> <privkey> <prefix>
```

- **Trace handle:**

Simple tool for resolving a handle.

```
java -cp handle.jar net.handle.apps.simple.HDLTrace <handle>
```

- **Home Prefixes:**

Simple tool for homing Prefixes. It uses public key authentication.

```
java -cp handle.jar net.handle.apps.simple.HomeNA <auth hdl>
<privkey> <server ip> <NA handle>
```

11 Configuring an Independent Handle Service

An independent or private handle service (such as a service maintained behind a firewall that is not publicly accessible) operates without contacting the Global Handle Registry. Configuring an independent service requires changes to the client for resolution to occur, and to the server for enabling authentication, homing and administrative tasks to be performed without the GHR.

Resolution Service Providers who wish to operate an independent handle service must so notify the Handle System Administrator in advance.

11.1 Client Configuration Details

This section explains how to configure the java client software to resolve handles locally, either through a resolution/caching server, or by directing specific prefixes to a certain service/site.

To specify a local handle server that should be used to process all resolution requests, follow these instructions:

Copy the `siteinfo.bin` file that describes the site/server where all resolution should be performed into a file called `resolver_site` in the `.handle` sub-directory of the users "home" directory. This will cause all non-administrative requests to be sent through the site described by `siteinfo.bin`. This should make resolution faster for organizations that can use the resolution server as a shared cache.

By default, no administrative messages are sent through this site (because administration must be done directly with the site that is responsible for each prefix and cannot be "tunneled".)

To force all messages (including administration messages) to go to the local resolution server described above, the user must specify the prefixes that are "homed" on the resolution server. All other prefixes will bypass the local resolution server. To specify the prefixes, do the following:

Create a file called `local_nas` in the `.handle` sub-directory of the users home directory. This file should contain one prefix handle on each line (e.g., "0.NA/11234"), encoded in UTF8 (ASCII is OK as long as there are no special characters). Every request for a handle having a prefix contained in this file will be sent to the local resolution site.

If no `resolver_site` file is provided, the `local_nas` file is ignore.

If there is more than one local handle gateway the methods described in the previous sections will not work. In this case a `local_info` file must be placed in the `.handle` directory on each local client machine. This file is prepared using a special utility that can be invoked using the command:

```
jre -cp handle.jar net.handle.apps.privatelhs.LocalInfoJPanel
```

This tool allows creation of a `local_info` file by creating a list of local handle gateways. For each gateway, use the "Load From File" button in the tool to import the `siteinfo.bin` file from the gateway's handle server directory. Once the site information is loaded, use the "Add" button in the @var{Naming Authorities section to create a list of prefixes this gateway should be responsible for.

Once the site information and naming authority list has been set for each local handle gateway, you can use the "Save to File" button on the main window to save your `local_info` file.

11.2 Server Side Configuration

By default, authentication, homing, and administering handles on a handle server require your handle server to communicate with the Global Handle Registry. This section describes how to configure your handle server so that administration of handles can be done without communicating with the Global Handle Registry.

(1) Modify the `config.dct` file:

```
"server_admin_full_access" = "y"
"allow_na_admins" = "no"
```

(2) To home a prefix on your handle server without contacting the Global Handle Registry, add the prefix to the `nas.jdb` file using the DBTool (See Chapter 7.1, *DBTool*).

(3) Once a prefix has been homed, create a new admin handle for it. (The default admin handle is the prefix itself. This default value cannot be used because it requires communication with the Global Handle Registry.) Create the new admin handle using the DBTool, and associate a secret key (password) with it at index 300. For example, if your prefix is 1234, add 0.NA/1234 to the `nas.jdb` file using the DBTool, then create the new admin handle 1234/ADMIN (use upper case when using the DBTool) with a secret key at index 300.

(4) Edit the `config.dct` file to change the "server_admins" entry to the new admin handle.

(5) Restart the server.