

ODBC Fetch

Administrator's Guide

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express permission of Autonomy Systems Ltd.

Windows is a trademark of Microsoft Corp.

Unix is a trademark of X/Open Ltd.

Copyright © 2001 Autonomy Systems Ltd.

ODBC Fetch is a trademark of Autonomy Systems Ltd.

Table of Contents

1. Introduction	1
How it works	1
Requirements	2
2. Installing ODBC Fetch	3
Preparing for the installation	3
Installing ODBC Fetch under Windows NT and Windows 2000	4
Installing ODBC Fetch under UNIX	6
Files in the ODBC Fetch	8
3. Configuring ODBC Fetch	11
Database Access	11
Typical Configurations	11
Available Parameters	21
Individual Fetch Configuration Files	23
Extracting Fields as a file	26
4. ODBC Content CGI	27
Configuration	27
5. ODBC Fetch Tutorial	29
The Database to retrieve from	29
The ODCB Fetch Configuration File	30
The Fetch Specific Configuration File	30
Templates	33
Scripts Directory Configuration Files and Templates	35
6. ODBC Fetch Troubleshooting	37
7. ODBC Fetch FAQ	38
Appendix A: Import Parameters	39
Appendix B: DRE Security Modules	69
Lotus Notes Security – Unmapped	77
Lotus Notes Security – Mapped	81
ODBC/ORACLE Security - Unmapped	85
Microsoft NT Security – Unmapped	88
Microsoft NT Security - Mapped	90
Microsoft Exchange Security – Unmapped	97
Exchange Security - Mapped	102
Appendix C: Service Settings for use with Autonomy DiSH	107

Autonomy

Autonomy employs a fundamentally different and unique combination of technology to enable computers to form an understanding of a page of text, web pages, e-mails, voice, documents and people.

Autonomy's solution is therefore able to power any application dependent upon unstructured information within every market sector, including: e-commerce, customer relationship management, knowledge management, enterprise information portals and online publishing applications.

This is evidenced by the significant penetration of the technology in a diversity of vertical markets and has been achieved principally because every market sector needs to manage and leverage the benefits of unstructured information.

Autonomy was founded in 1996 and has offices in Boston, Chicago, Dallas, San Francisco, New York, and Washington, D.C. in the United States, as well as offices through Europe, including Amsterdam, Brussels, Cambridge, Frankfurt, Milan, Paris, Oslo, and Sydney. In July 1998, the company went public on the EASDAQ exchange (EASDAQ:AUTN). Autonomy floated on The NASDAQ National Market (NASDAQ: AUTN) in May 2000, and on the London Stock Exchange (LSE: AU) in November 2000.

To contact Autonomy, please get in touch with your nearest location listed below.

Europe and South Pacific

Autonomy Systems Ltd.
Cambridge Business Park
Cowley Road
Cambridge
CB4 0WZ

Help Desk: +44 (0) 800 0 282 858
Switchboard: +44 (0) 1223 448 000
Fax: +44 (0) 1223 448 001
E-mail for information: autonomy@autonomy.com
for support: uksupport@autonomy.com

The Help Desk operates from 9.30 am to 6.00 pm (GMT) Monday to Friday.
Website: www.autonomy.com

USA

Autonomy Inc.
301 Howard Street
22nd Floor
San Francisco
CA 94105

Help Desk: +1 877 333 7744
Switchboard: +1 415 243 9955
Fax: +1 415 243 9984
E-mail for information: info@us.autonomy.com
for support: support@us.autonomy.com

The Help Desk operates from 9.30 am to 6.00 pm (CST) Monday to Friday, toll-free.
Website: www.autonomy.com

Welcome

Thank you for choosing Autonomy and welcome to your ODBC Fetch™ version 2.2 Administrator's Guide.

Autonomy Solutions

Autonomy solutions are built in a modular architecture, based on the Autonomy Application Builder™ (API). They provide an infrastructure that is fully scalable and customizable according to customers' present and future needs. Autonomy solutions include:

- **Portal-in-a-Box™**, our comprehensive and fully-automated Information Portal for content-rich Internet and Intranet sites,
- **ActiveKnowledge™**, which conducts a real-time analysis of the ideas involved in the content of any opened application and provides real-time links to relevant internal and external information,
- **i-WAP™**, an add-on to Portal-in-a-Box™, automating the delivery and personalization of timely and relevant information to mobile users.
- **Autonomy Server™** provides a fully automated and precise means of retrieving information. It allows content to be queried in any language and any format, wherever it is stored, and presented with hyperlinks to similar information, automatically and in real-time.
- **Autonomy Update™** automatically creates a personalized report, informing individual users of developments that are relevant to their specific roles and interests. The alert capability enables users to keep track of late-breaking news, automatically and in real-time.
- **Autonomy Application Builder™** is a toolkit that enables companies and partners to customize Autonomy's products according to their individual requirements. The Application Programming Interfaces (APIs) have been modularized into functional suites to allow developers to plug in only those modules required for a given application. The APIs are distributed with the Autonomy Server™ and a set of sample code.

1. Introduction

ODBC Fetch is a powerful "fetching" solution allowing information to be gathered from ODBC data sources and indexed in to a DRE provided in the installations of the Knowledge Server or Knowledge Update. ODBC Fetch can retrieve content as text directly, use the content as a pointer to a local filename or automatically retrieve documents in a variety of formats from fields of an ODBC database.

Each instance of ODBC Fetch can fetch from multiple ODBC data sources simultaneously. The documents fetched are automatically imported into Autonomy's indexing file format and indexed in to a Dynamic Reasoning Engine (the core technology in the Knowledge Server or Knowledge Update).

The entire process - retrieving data, importing and indexing is referred to as a **fetch**.

The ODBC Content Retrieving CGI is also supplied with the ODBC Fetch. This CGI can be used in conjunction with the Knowledge Server or Knowledge Update to allow content retrieved from ODBC data sources to be returned in HTML format. Together the ODBC Fetch and ODBC Content Retrieving CGI provide the complete solution to extraction and display of data from ODBC databases.

How it works

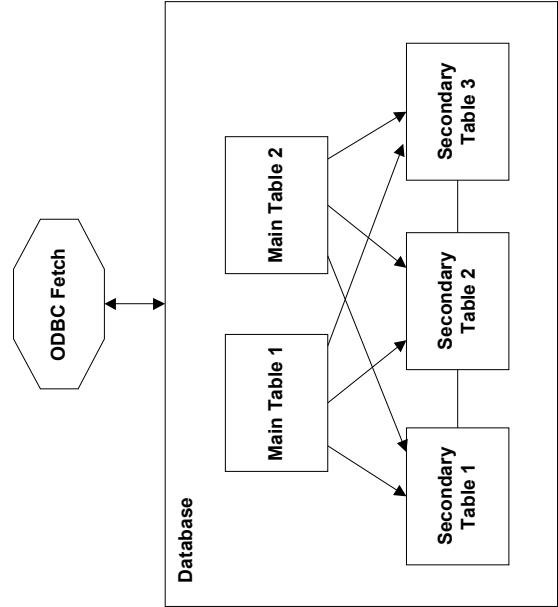
ODBC Fetch uses an ODBC data source to gather content from a database. The tables or views that are to be fetched must be specified, together with the fields in that table which contain the content that is to be indexed into a DRE. Configuration files specify how tables are related to one another and template files define how that data is stored.

Each fetch has a main table where each row in this table represents a document in the DRE. Secondary tables contain additional information for each document. Each row is read from the database and put into an index file for the DRE to read. This index file is then indexed into the engine.

During the fetch the individual documents are given a unique reference which specifies the row of the table from which it derived. This reference is prefixed by the IP Address and location of the ODBC Content Retrieving CGI, meaning; when a document originating from an ODBC database is returned as a link in Knowledge Update or Knowledge Server, clicking on the link will automatically pass control to the ODBC Content Retrieving CGI, which in turn returns the results to the user in its original format.

Introduction

If the fetch that retrieved the information was configured to store the content in the DRE then the content may also be served back as plain text in the normal way, with the loss of its original formatting but a decrease in time taken to return the content.



Requirements

ODBC version 2.0 (or higher) drivers for the target database type(s). The Server also needs to have an appropriate Datasource set up for each database containing content. This is done via the Control Panel. Datasources should be configured as System-wide (the "System DSN" tab) to ensure that they are accessible to all users.

The ODBC Fetch requires an underlying ODBC Driver in order to connect to the data source. For example, if you want to run the ODBC Fetch on Solaris to access an Oracle 8.0 server running on the same or another platform, you must acquire an Oracle 8.0 compatible ODBC Driver for Solaris.

2. Installing ODBC Fetch

Preparing for the installation

Before installing ODBC Fetch it is advisable to have the following information ready:

Information	Description
Installation Name	If you are licensed to have multiple installations, then each installation requires a distinct name to differentiate it from other installations.
License Holder Name	The name of the person or company to which the product is licensed.
Autonomy Query Engine settings	ODBC Fetch will automatically index documents into your Knowledge Server or Knowledge Update. You require the settings for the DRE you wish to index the data into, these include: the IP Address of the DRE machine, the Query and Index Port (this can be found listed in the DRE.INI file in the main directory of your DRE), the Database you wish to index the documents into (again, the possible databases are listed in your DRE.INI file)
Indexing Details	ODBC Fetch uses a file to store indexing information; the DRE is then instructed to index in this file. You need to decide where you wish these files to reside, and you will need to know the full path of the directory from the ODBC Fetch and from the engine.
ODBC database details	Which ODBC database from which to obtain the data.

Installing ODBC Fetch under Windows NT and Windows 2000

Note: ODBC Fetch should be installed by the system administrator.

To install under Windows insert the ODBC Fetch CD-ROM into your CD-ROM Drive.

Start the installation by double-clicking on the ODBCFetch-2.2.0.exe program in the root directory of the CD-ROM through Windows explorer.

Read and follow all installation instructions on the screen carefully. Before the installation program can start to copy files onto your PC, you need to provide it with some information:

1. Read the license agreement and click on **Next** to accept it.
2. If you are installing the ODBC Fetch 30 day evaluation version, enter the license holder.
3. Enter a unique name for the ODBC Fetch installation.

Note: The unique name must not contain any spaces.

4. Select the directory in which you want to install ODBC Fetch. By default it is installed on **C:\Autonomy\ODBCFetch**, but you can use the **Browse** button to navigate to another location.
5. Enter the web server details where the ODBC fetch is to be installed as follows:

Scripts directory mapping

The directory to which the web scripts are mapped. By default this is **/Scripts**.

IP Address

The IP address of the machine on which you want to install the Fetch.

6. Enter the details of the DRE into which the Fetch should index the documents that it fetches:

IP Address

The IP address of the machine on that the DRE is running.

Query Port

The port that is used to send queries to the DRE.

Index Port

The port that ODBC Fetch uses to index information into the DRE.

7. Enter the name of the DRE database into which ODBC Fetch should index documents.

8. If you have DiSH installed, enter the service port that it will use to control the ODBC Fetch.
9. Specify the location of the index files that ODBC Fetch generates:

Fetch creates index files in:

The path to the location where ODBC Fetch creates index files before they are indexed.
By default this is **C:\Autonomy\ODBCFetch**.

DRE reads index files from:

The path to the location from which the DRE reads the index files. By default this is
C:\Autonomy\ODBCFetch.

10. Specify whether you want to add shortcuts to the Windows Start menu. If you do, select the name of the Program Manager group to which you want to add the ODBC icons.

The installation program now has all the information it requires. Click on **Next** to start the installation of ODBC Fetch. You are notified when the installation process is completed.

Installing ODBC Fetch under UNIX

Note: ODBC Fetch should be installed by the system administrator.

To install under Unix insert the ODBC Fetch CD-ROM into your CD-ROM Drive. Read and follow all installation instructions on the screen carefully.

1. Copy the ODBC Fetch installer from the CD to your local disk.
2. Uncompress the installer using the command:
`uncompress <Installer>.tar.z`
3. Un-tar the resulting file using the command:
`tar -xvf <Installer>.tar`

This extracts the following files into the <Installer> directory:

```
odbcfetch
Setup.sh
freplace.exe
binslave.cfg
binslave.exe
freplace
importslave.exe
phraselist.dat
phraselist_doc.dat
phraselist_ppt.dat
phraselist_qxd.dat
ExampleTable1.tmpl
ExampleTable2.tmpl
ExampleTable3.tmpl
StartODBCFetch.sh
StopODBCFetch.sh
Uninstaller.sh
example.cfg
files.txt
```

```
odbcoci.cfg
odbcoci.sh
odbcfetch.cfg
odbcoci
```

4. You next need to run the installer script, `Setup.sh` with the following arguments:

<Base Name>

A unique name for the ODBC Fetch installation.

Note: The unique name must not contain any spaces

<Application directory (fully qualified path)>

The directory in which you want to install ODBC Fetch.

<DRE Host>

The IP address of the machine on that the DRE is running.

<DRE index port>

The port that ODBC Fetch uses to index information into the DRE.

<DRE database >

The DRE database into which the ODBC Fetch indexes the data.

<Scripts directory>

The path to the location of the scripts directory for your web server.

For example:

```
./Setup.sh MyODBCFetch /autonomy/ODBCFetch 127.0.0.1 2500 2501 News
/opt/ns-home/cgi-bin/
```

5. Check that the settings, which you have entered, are correct and press Return in order to complete the installation.
6. You are now ready to modify the ODBC configuration file (the configuration file is located in the directory in which you installed ODBC Fetch).

Files in the ODBC Fetch

The ODBC Fetch has one main configuration file, ODBCFetch.cfg, which governs the overall behavior of the Fetch. All the fetches that need to run are specified here however each Fetch has its own configuration file, <Fetchname>.cfg, which governs the overall behavior of that Fetch.

NT

- In the Installation directory

\ODBCFetch

ODBCFetch.cfg	(Fetch main configuration file)
ODBCFetch.exe	(Fetch application executable)
(Individual Fetch configuration files)	
(Uninstaller and utility files)	
(Various table templates)	(Templates used to create the idx file to index into the DRE)

- In the Scripts directory of the Web Server

\Scripts

ODBCFetch.exe	(Fetch CGI executable)
ODBCFetch.cfg	(Fetch CGI Configuration file)
(Individual Fetch CGI configuration files)	
(Various table templates)	(Templates used to generate the html files, which are used to display the Database document)

Unix

- In the Installation directory

\ODBCFetch

- ODBCFetch.cfg (*Fetch main configuration file*)
- ODBCFetch.exe (*Fetch application executable*)
- (Individual Fetch configuration files)
- (Uninstaller and utility files)
- (Various table templates) (*Templates used to create the ndx file to index into the DRE*)

- In the Scripts directory of the Web Server

\Scripts

- ODBCFetch.exe (*Fetch CGI executable*)
- ODBCFetch.cfg (*Fetch CGI Configuration file*)
- (Individual Fetch CGI configuration files)
- (Various table templates) (*Templates used to generate the html files, which are used to display the Database document*)

3. Configuring ODBC Fetch

This Chapter describes how to add to and change the configuration of an installed ODBC Fetch.

Please be aware that any key=value parameter that calls a value from else where, the value setting MUST be treated as case sensitive.

In all cases it must be noted that new configuration settings will only take effect when the ODBC Fetch service is stopped and restarted.

Database Access

Database Accounts

The majority of database servers require some sort of user account and password before access to data is allowed. It is often most convenient to explicitly set up an account on the database for sole use by Autonomy products, although this is by no means a requirement. The name and password of the account under which access is to occur should be noted – there is an option to encrypt these if so desired which is discussed below. The account used for access must have the necessary permissions to view the tables that are to be fetched -- these are often called "Select" permissions although the names and details of setting these up vary between database vendors.

Typical Configurations

The typical installation of ODBC Fetch consists of a single instance running on a single machine. In this case the parameters used for the fetch procedure are stored in a file residing in the same directory as the ODBC Fetch application executable. The configuration file will have the same name as the executable, but the extension ".cfg" instead of ".exe". Each fetch has a separate configuration file to define which tables it should fetch.

Structure of Example Tables

The example below outlines a single fetch configuration which fetches three tables, ExampleTable1, ExampleTable2 and ExampleTable3. ExampleTable1 is the main table and one row in this table will represent one document in the DRE. There is a one to many relationship from ExampleTable1 to the other two tables defined by the key field, ID. The three tables have the following structure:

Field Name	Description
ID	Primary key. In the main table, ExampleTable1, this uniquely identifies a row. In the other two tables it defines which rows correspond to a row in the main table.
TitleField	This is the title of the row and will be used as the document title in the DRE.
ContentField	This field contains the content for the document.

ID	Example Table 1
1	Smith
2	Jones
3	Bloggs

ID	Example Table 2
1	Socks
1	Trousers
2	Skirt
3	Pants

ID	Example Table 3
1	01/05/2000
1	27/10/2000
2	17/09/2001
3	16/08/2001

As you can see above, Example Table 1 is the main primary table with all ID numbers unique, and Example Table 2 and 3 are Secondary Tables with additional content connected by the ID field. Therefore, when the DRE document is constructed, each document will look as follows:

ID 1
Smith
Socks
Trousers

ID 2
Jones
Skirt
01/05/2000
27/10/2000

ID 3
Bloggs
Pants
17/9/2001
16/8/2001

An Example Configuration

This is the main configuration file:

Configuration File	Comments
[License]	Do not remove this section, it holds important license information.
Holder=MyCompany	The name of the licensed party
Key=XXX54749XX34794XX	The license key (If you need to extend your license you can obtain a new key)
[Default]	This section contains parameters that apply to all fetches
DreHost=123.456.789.111	The IP Address of the engine into which you wish the documents to be indexed
DREIndexPort=1234	The Index port number of the engine.
Database=MyODBCData	The database within the engine into which the documents should be indexed
FetchStartTime=23:59	Start the fetch just before midnight each night. NB for midnight use 24:00 not 00:00.
FetchRepeatInterval=86400	Run the fetch once a day.
FetchCycles=-1	Fetch for an unlimited amount of days.
[Fetch]	This section defines which fetches are used.
Number=1	The number of fetches used by this ODBCFetch
FetchName0=Example	The name of fetch 0
[Example]	This section details parameters for the fetch called 'Example'
Connection=DSN=Example;UID=user;PWD=password	This is the ODBC connection string. DSN is the datasource name that was defined in the control panel ODBC settings. UID and PWD and the user id and password required for a connection to the database.
Parser=Example.cfg	The configuration file this fetch is to use for table definitions.

Configuring ODBC Fetch

This is the individual configuration file for the fetch "Example". It defines three tables, ExampleTable1, ExampleTable2 and ExampleTable3. The main table is ExampleTable1 and the other two are related to this by a field called ID.

Configuration File	Comments
[Configuration]	
NumTables=3	The number of tables this fetch is going to use.
TableName0=ExampleTable1	First table.
TableName1=ExampleTable2	Second table.
TableName2=ExampleTable3	Third table.
Data Type=Example	Each row in the main table will be written to a temporary file with the prefix example.
BaseDirectory=Example	This fetch will use a directory called Example to store temporary files in, generally it is best to use the spider name.
PostProcess=false	This feature is disabled here but allows documents to be ignored if certain fields are blank.
SingleFile=false	The index file can either be generated directly from the templates or, in this case, for each record a temporary file is generated and these are processed to produce an index file. This method is slightly less efficient but more flexible as it allows documents to be ignored if they do not meet certain criteria (e.g. They must contain a minimum number of words).
Extension=.htm	The temporary files generated are in ndx format. This is a single document in Autonomy's index file format. They could, for example, be in html format and the html saved so content did not have to be retrieved from the database each time if desired.
MainTable=ExampleTable1	The main table. Each row in this table represents a single document.
[ExampleTable1]	This section details the main table ExampleTable1

Configuration File	Comments
Target=c:\Examples.mdb..ExampleTable1	<p>This is the table in the actual database which this table maps to. Note that there are several options as follows:</p> <ol style="list-style-type: none"> 1) Target=<table name> 2) Target=<database name>.<table name> 3) Target=<filename of database>.<table name> <p>For all databases except ms access For access databases only (must have two dots).</p> <p>In the first case, the default database will be used. In the second and third cases, the specified database will be used.</p> <p>E.g. to specify a database</p> <p>Others: Target=MyDatabase.MyTable</p> <p>Access: Target=MyDatabase.mdb.MyTable</p> <p>Note that when using Access, spaces in the path will not work</p>
PrimaryKeyFields=1	There is one primary key field in this table. Each table must contain at least one primary key field. In the main table, the combination of all primary key fields must uniquely identify a row.
PrimaryKeyName0=ID	The name of the primary key field in the database.
PrimaryKeyType0=Unquoted	Fields which can contain spaces in their values should be quoted.
Template=ExampleTable1.tmpl	The template file to use for this table.
ForeignKeys=2	There are two foreign keys in this table. A foreign key specifies the rows in another table which correspond to a row in this table and can be made up of one or more fields.
ForeignKeyFields0=1	The first foreign key is made up of one field.
ForeignKeyName0_0=ID	The field is called ID.

Configuring ODBC Fetch

Configuration File	Comments
ForeignKeyType0_0=Unquoted	As PrimaryKeyType.
ForeignKeyTable0=ExampleTable2	This foreign key relates to PrimaryKey0 in ExampleTable2
ForeignKeyFields1=1	The second foreign key also consists of one field
ForeignKeyFields1_0=ID	The field is called ID
ForeignKeyType1_0=Unquoted	As PrimaryKeyType
ForeignKeyTable1=ExampleTable3	This foreign key relates to PrimaryKey0 in ExampleTable3
[ExampleTable2]	This section details the secondary table ExampleTable2
Target=c:\Examples.mdb..ExampleTable2	This is the table in the actual database which this table maps to. See above.
PrimaryKeyFields=1	There is one primary key field in this table. In secondary tables, the combination of all primary key fields need not uniquely identify a row.
PrimaryKeyName0=ID	The name of the primary key field in the database.
PrimaryKeyType0=Unquoted	Fields which can contain spaces in their values should be quoted.
Template=ExampleTable2.tmpl	The template file to use for this table.
[ExampleTable3]	This section details the secondary table ExampleTable3
Target=c:\Examples.mdb..ExampleTable3	This is the table in the actual database which this table maps to. See above.
PrimaryKeyFields=1	There is one primary key field in this table.
PrimaryKeyName0=ID	The name of the primary key field in the database.
PrimaryKeyType0=Unquoted	Fields which can contain spaces in their values should be quoted.
Template=ExampleTable3.tmpl	The template file to use for this table.

Template Files

The ODBC Fetch uses template files to define the layout of the extracted data. In this example it will generate an htm file for each record with the following format:

```
<HTML><HEAD>
<TITLE><!--TITLEFIELD--></TITLE>
<META NAME="DRREFERENCE"
CONTENT="http://127.0.0.1/Scripts/ODBCFetch.exe?METHOD=FetchData&KEYID=2&DOC
TYPE=.html">
</HEAD>
<BODY>
<!--CONTENTFIELD--><BR>
<BR>
Table 2 <BR>
Title 2: <!--TITLEFIELD--><BR>Content 2: <!--CONTENTFIELD-->
<BR>
<BR>
Table 3 <BR>
Title 3: <!--TITLEFIELD--><BR>Content 3: <!--CONTENTFIELD-->
<BR>
</BODY>
</HTML>
```

The template files are text files with tags in them. A tag is either replaced with field contents or the data from another table, which is generated with a template file in the same way. A tag to be replaced with field contents takes the form <!--FieldName-->. A tag which is to be replaced with data from another table takes the form <!--link.TableName-->. Note that TableName is the name of the table in the ODBC Fetch configuration file, not the name in the database.

Configuring ODBC Fetch

The template files will contain the following:

ExampleTable1.tmpl:

```
<HTML>
<HEAD>
<TITLE><!--TITLEFIELD--></TITLE>
<META NAME="DRREFERENCE"
CONTENT="http://127.0.0.1/Scripts/ODBCFetch.exe?METHOD=FetchData&KEYID=<!--
ID-->&DOCTYPE=.html">
</HEAD>
<BODY>
<!--CONTENTFIELD--><BR>
<BR>
Table 2<BR>
<!--link.ExampleTable2--><BR>
<BR>
Table 3<BR>
<!--link.ExampleTable3--><BR>
</BODY>
</HTML>
```

Example Table2.tmpl:

```
Title 2: <!--TITLEFIELD--><BR>
Content 2: <!--CONTENTFIELD-->
```

Example Table3.tmpl:

```
Title 3: <!--TITLEFIELD--><BR>
Content 3: <!--CONTENTFIELD-->
```

The tmpl files in the cgi directory are the format of the html page to generate. The tmpl files that appear in the ODBC directory must contain the DRREFERENCE line, and within the body tag should be the fields that are to be indexed into the DRE. Note that the line

```
<META NAME="DRREFERENCE"
CONTENT="http://127.0.0.1/Scripts/ODBCFetch.exe?METHOD=FetchData&KEYID=
```

Configuring ODBC Fetch

The ODBCFetchCGI

The DRREFERENCE is normally a URL to display the document. In this case we will use the ODBC Fetch CGI which extracts data in much the same way as the ODBC Fetch. The configuration file defining the tables is the same as for the ODBCFetch. The main configuration file consists of only two options in the [configuration] section:

```
[Configuration]
Parser=C:\Inetpub\Scripts\Example.cfg
Connection=DSN=Example;UID=;PWD=
```

Parser specifies the configuration file to use for table structure and ODBC specifies the ODBC connection string. These are the same as in the main ODBCFetch.

The CGI arguments are:

```
Method=FetchData
Key<Key name>=<Value to identify a row>
```

Available Parameters

This section details the possible parameters, provides descriptions, default values, the range of possible values and specifies which sections the parameter can be used in.

Parameter	Default	Range	Description	Section(s)
Holder	NA	NA	The License holder	License
Key	NA	NA	The License key	License
FetchCycles	Infinite	-1 -	Number of times to repeat (-1 for infinite)	Default, Individual Fetch
FetchRepeatSecs	86400	1-	Number of seconds between each cycle	Default, Individual Fetch
FetchStartTime	Current time	HH:MM	Time at which to start the fetch.	Default, Individual fetch
DreHost	NA	Any IP	The engine (DRE) machine's IP address	Default, Individual Fetch
DREIndexPort	NA	Any port	The engine (DRE) indexing command port	Default, Individual Fetch
IndexPath	NA	Any directory	The path to index files relative to the DRE	Default, Individual Fetch

Configuring ODBC Fetch

Parameter	Default	Range	Description	Section(s)
IndexOverSocket	False	True/false	If true, data will be indexed over a socket, otherwise the DRE will read the index file directly from the disk. Using this setting will preserve the IDX file created.	Default, Individual Fetch
ImportPath	NA	Any directory	The path to index files relative to the ODBC Fetch	Default, Individual Fetch
FetchName0N	NA	Any string	Name of configured fetches	Fetch
Connection	NA	ODBC Connection string	The string to use when connected to an ODBC Data source	Default, Individual fetch
Parser	NA	Any file	The configuration file defining table structure. The "parser" is one-to-one, i.e. each configuration file can only handle one parser, and hence a separate EXE is required for more than one.	Default, Individual Fetch
Database	Default	NA	The database to use in the engine	Default/ Fetch
FixedFieldName0..X	NA	Any String	Specify the name of a fixed value field to be added to retrieved documents indexed into the DRE	Default/ Fetch

Parameter	Default	Range	Description	Section(s)
FixedFieldValue0..X	NA	Any String	Specify the value of a fixed value field to be added to retrieved documents indexed into the DRE	Default/ Fetch
FetchCleanFiles=	false	True/false	This clears the IDX files and the files that it has fetched on startup. This means when the fetch is stopped and restarted NOT on a new cycle.	default

Individual Fetch Configuration Files

Parameter	Default	Range	Description	Section(s)
NumTables	0	0-	Number of tables that this fetch is going to use.	Configurati on
TableName0..N	NA	Any string	Name of each table. The name specifies which configuration section to use, not the name of the table in the database.	Configurati on
Datatype	NA	Any valid filename prefix	The prefix to use for temporary files generated when SingleFile=false	Configurati on
Extension	NA	Any file extension	The extension to use for temporary files	Configurati on

Configuring ODBC Fetch

Parameter	Default	Range	Description	Section(s)
Connection	NA	Any ODBC Connection string	The string used to connect to the ODBC data source. This is of the form DSN=<Datasource name>;UID=user;PWD=password	Configuration
MainTable	NA	Any table name	The name of the configuration section for the main table. One DRE document represents one row in this table.	Individual Table
Target	NA	Any table name	The name of the table in the database	Individual Table
PrimaryKeyFields	NA	1 -	The number of primary key fields.	Individual Table
PrimaryKeyName0..N	NA	Any field name	The name of a primary key field	Individual Table
PrimaryKeyType0..N	NA	Unquoted/Quoted	Whether to put quotes around a field value.	Individual Table
Template	NA	Any filename	The template file for this table	Individual Table
ForeignKeys	NA	0-	The number of keys relating to other tables. Note that a key can consist of more than one field.	Individual Table

Parameter	Default	Range	Description	Section(s)
ForeignKeyName0..N_0..N	NA	Any field name	This specifies the name of a field in a foreign key. The first number in the key name indicates which foreign key this field is in. The second is the number of that field in the foreign key which maps to the number of the PrimaryKey in the secondary table.	Individual Table
ForeignKeyType0..N_0..N	NA	Quoted/Unquoted	As PrimaryKey0..N_0..N	Individual Table
Where	NA	Any string	Specifies a where clause to restrict the rowset. E.g. Department=Admin	Individual Table
OrderBy	NA	Any string	Specifies the order in which the rows of a table are fetched	Individual Table
Select	NA	Any string	Restricts the fields which are retrieved from a table	Individual Table
SingleFile	False	Boolean	Specifies whether or not idx formats for each document are to be appended to a single idx file.	Individual Table
Outputfile		.idx	Specifies the name of the single idx file that is created when SingleFile=true	Individual Table

For Import Parameters see Appendix.

NB: ODBC Content Retrieving CGI is of the form

<http://ip.address/scripts/<ODBCContentServerName>.exe>

This link will be located in the scripts directory probably called cgi-bin on non-Microsoft Servers. Since it forms the start of the reference, clicking on it in the Knowledge Server or Knowledge Update will fire up the CGI and display the content.

Extracting Fields as a file

It is possible using the ODBC Fetch to extract a field as a file. For example, if you have ms-word documents stored in a field called MyField you can extract them by setting the following parameters in the corresponding files:

In the configuration file for the individual fetches:

```
BlobField=<MyField>
BlobExtensionFromField=False
BlobExtension=.doc
Extension=.idx
```

The template file for the main table would look something like this:

```
#DRREFERENCE http://10.1.1.10/ScriptODBCFetchblob.exe?METHOD=FetchData&KEYID=<!--ID-->
&DOCTYPE=.doc
#DRETITLE blobtext<!--id-->
#DREENDDOC
```

The cgi also takes an optional extra argument, DOCTYPE. This specifies the file extension of the document. The cgi configuration file also has a list of file extensions and mime type mappings as follows:

```
DocType0=.htm
MimeType0=text/html
DocType1=.html
MimeType1=text/html
DocType2=.txt
MimeType2=text/html
DocType3=.doc
MimeType3=application/msword
```

4. ODBC Content CGI

The ODBC Content CGI is a CGI program, which allows other Autonomy server products such as the Knowledge Update, and the Knowledge Server to return ODBC derived information in its native format. When a document is retrieved from an ODBC database it is given a reference which reflects the location of the ODBC Content CGI program *and* encodes the table and row from which the information originally derived. The references returned in the results list of both Knowledge Update and Knowledge Server take the form of hypertext links -- when the information has been extracted via ODBC Fetch clicking this link passes control to the ODBC Content CGI which displays the information in its native format.

Configuration

The manner in which content is returned is configurable -- the following describes the parameters that are required. Data sources for each database that the ODBC Content CGI will be expected to use must be configured in a similar manner to section (3.3). The ODBC Content CGI requires a configuration file in the same directory as itself with the extension .cfg. This configuration file contains only the following two options:

```
[Configuration]
Parser=C:\Inetpub\Scripts\Example.cfg
Connection=DSN=Example;UID=;PWD=
```

Parser specifies the configuration file to use for table structure and ODBC specifies the ODBC connection string. These are the same as in the main ODBCFetch.

The CGI arguments are:

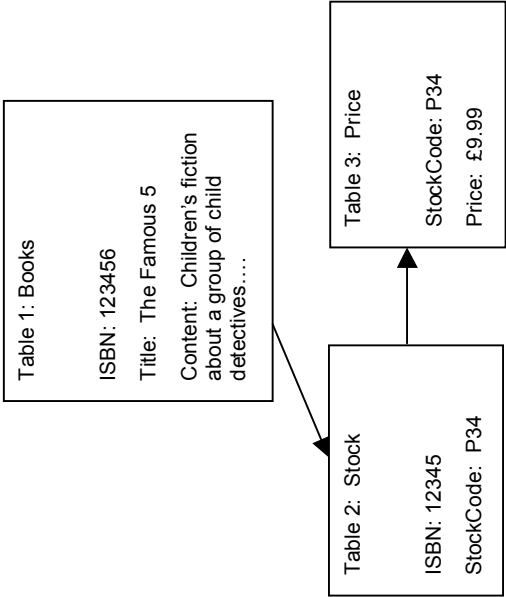
```
Method=FetchData
Key<Key name>=<Value to identify a row>
```


5. ODBC Fetch Tutorial

This section will take you through step by step as to how one should set up the ODBC Fetch.

The Database to retrieve from

We have a database with 3 tables:



If we want to extract all the data we need to do the following:
Set up an ODBC Driver (called Example here).

The ODCB Fetch Configuration File

Set up the C:\ODBCFetch\ODBCFetch.cfg as follows:

1. under the [Fetches] section add the following:
2. Number=1
3. FetchName0=Example
4. Under the [Example] section in the script file, set the following:
5. Connection=DSN=example;UID=;PWD=
6. Parser=Example.cfg (we will change this file next)

The Fetch Specific Configuration File

Set up the C:\ODBCFetch\Example.cfg as follows:

NumTables=The amount of tables in the database. (I.E. NumTables=3)

TableName0= *MyTable* (This can be anything but for simplicity sake label each by table name)
(I.E. TableName0=Books)

TableName1= MyNextTable (I.E. TableName1=Stock)

TableName2=etc...(I.E. TableName2=Price)

E.g. NumTables=3

TableName0=Books

TableName1=Stock

TableName2=Price

7. Set MainTable=Books

8. Having completed the [Configuration] section please find the [ExampleTable1] section. Change the section name to read the name of your first table, beneath that you need to fill in the following:

Target=*this is looking for the actual table name within the database.* (i.e. Target=Books)
 PrimaryKeyFields=*The Number of primary key fields in this table* (i.e. PrimaryKeyFields=1)
 PrimaryKeyName0=*Name of the Primary Key Field in this table* (i.e. PrimaryKeyName=ISBN)
 PrimaryKeyType0=Unquoted (leave this default, it means there can be spaces in the field)
 ForeignKeys=*Number of foreign key fields in this table (links to other tables to be indexed)* (i.e. ForeignKeys=1)

//Show for this table the relationship to the next table

ForeignKeyFields0=*The number of foreign key fields in this table* (i.e. ForeignKeyFields0=1)
 ForeignKeyName0_0=*First field linked to other tables* (i.e. ForeignKeyName0_0=ISBN)
 ForeignKeyType0_0=Unquoted **leave as default**
 ForeignKeyTable0=*Table name linked by foreign key field* (i.e. ForeignKeyTable0=Stock)

Template=Books.tmpl

E.g.

[Books]

Target=Books

PrimaryKeyFields=1

PrimaryKeyName0=ISBN

PrimaryKeyType0=Unquoted

Template=Books.tmpl

ForeignKeyFields=1

ODBC Fetch Tutorial

```
//Relating to Stock table
ForeignKeyFields0=1
ForeignKeyName0_0=ISBN
ForeignKeyType0_0=Unquoted leave as default
ForeignKeyTable0=Stock
```

Set up a section for [Stock] linking to the Price database and then a [Price] section, which does not link (so will have no Foreign Keys), set this up following the examples above.

```
E.g.
[Stock]
Target=Stock
PrimaryKeyFields=1
PrimaryKeyName0=ISBN
PrimaryKeyType0=Unquoted
ForeignKeyFields=1
```

```
//Relating to Price table
ForeignKeyFields0=1
ForeignKeyName0_0=ProductCode
ForeignKeyType0_0=Unquoted leave as default
ForeignKeyTable0=Price
```

```
Template=Stock.tmpl
```


Do the same for [Price]. This is all that needs to be in our [TableName] section, delete or comment out (//) all unused parameters.

```
[Price]
Target=Price
PrimaryKeyFields=1
PrimaryKeyName0=ProductCode
PrimaryKeyType0=Unquoted
ForeignKeyFields=0

Template=Price.tmpl
```

Templates

- Next we need to create a template for each of our tables to be indexed.
There are two types of tags which are used within the templates themselves:-

Field Tag - For each field in the table that needs indexing into the DRE you need to insert a tag `<!--FieldName-->`. When the temporary HTML documents is created each record, the tag will be replaced by the value for the specified field. For example: `<!--TITLEFIELD-->`.

Link Tag - allows you to display the contents of field(s) from another table within the database. For every table linked to this table via a foreign key, you need to insert another tag `<--link.TableName-->`. When the temporary HTML document is created for each record, this link tag will be replaced by the contents of the foreign table's HTML template. For example: `<--link.Stock-->` will insert the information from the stock table.

From the above example I would like to produce from this fetch an output document which looks like...

```
TITLE
CONTENT
AUTHOR

PRICE
```

ODBC Fetch Tutorial

ISBN NUMBER
STOCK CODE

Now to achieve this I have to build the templates, now bear in mind that all tables need a template file, and each table needs to be linked.

The main table will be used to link in the other tables with the information we require.

```
<HTML>
<HEAD>
<TITLE><!-- Title--></TITLE>
<META NAME="DREREFERENCE"
CONTENT="http://193.254.27.141/c:\inetpub\Scripts\ODBCFetch.exe?METHOD=FetchData&KEYISBN
=
```

Stock.tmpl we need to define that we are going to extract the ISBN and StockCode so it will look like..

***** Here we are going to pull through the information from

Price.tmpl *****

***** So in the final document here you will see PRICE

<!--link.Price-->

<!--ISBN-->

<!--StockCode-->

If we take Price.tmpl we just need to define that we are going to extract the PRICE so it will look like..

<!--PRICE-->

All these templates reside in the installed directory. All the above configuration files and templates deal with getting data into the DRE, now we need to look at being able to display it, getting data out of the DRE!

Scripts Directory Configuration Files and Templates

10. In the \Scripts directory there will be a set of .cfg files:

11. ODBCFetch.cfg

This lists a set of mime types but the bit we need to edit is as follows:

Parser=Example.cfg (we are going to edit this file next)

Connection=DSN=example;UID=;PWD= (For this example we will not use password protected files)

ODBC Fetch Tutorial

12. We need to set up the Example.cfg file in the \Scripts directory next, this should read the same as the previous Example.cfg file we did (I would just copy it across)
13. Now we need to change the templates within the scripts directory, this will show how the results will be displayed within the front end:

```
<HTML>
<HEAD>
<Title> Book Title: <!--Title--> </Title><BR>
</HEAD>
<BODY>
<b> Book Content:</b><!--Content--><BR>
<b> Author:</b> <!--Author--><BR>
<BR>
<!--link.Stock--><BR>
<BR>
</BODY>
</HTML>
```

This is the template for the Stock table.

```
<!--link.Price--><BR>
<BR>
<b> ISBN Number:</b><!--ISBN--><BR>
<b> Stock Code:</b> <!--StockCode--><BR>
<BR>
```

The template for the Price table is as follows:

```
<b> Price:</b><!--Price--><BR>
```

Having completed all these steps you should find that, having run the ODBC Fetch, the data you wanted to get into and out of the DRE is available.

6. ODBC Fetch Troubleshooting

This section describes a scenario in which the ODBC Fetch might not be functioning correctly. It should give the administrator an idea of the things to look out for when something seems to have gone wrong.

This section will assume that the administrator knows how to administer services on NT, etc.

The table below shows descriptions of the problem and quick guidelines on what to check.

PROBLEM	DESCRIPTION	ACTIONS
Documents are not being imported/indexed.	Documents are not appearing in the IDX file/DRE. The document is outside the document size criteria.	Check the ImportMinLength, ImportMaxLenght, ImportMinLengthWords ImportMaxLenghtWords settings. If these appear to be correct for the documents in question, enable import logging to see why the document is not being imported. The settings required are ImportLogFile=import.log ImportLogFileAppend=TRUE ImportLogLevel=FULL. This will produce a file, import.log, containing information about why documents are not being imported which will help track down the problem.

7. ODBC Fetch FAQ

This section lists a series of Frequently Asked Questions about the ODBCFetch.

Why do they use "numbers" to name the folders are created by the ODBC Fetch? Do they have any meaning?

It is just an Autonomy naming convention, we could have just had one folder, but this would have reduced the speed of the fetch, so we created many directories and numbering them seemed the easiest convention, they have no meaning.

Appendix A: Import Parameters

Please be aware that any key=value parameter that calls a value from else where, the value setting **MUST** be treated as case sensitive.

Input

Parameter	Default	Range	Description
HTMLImportExtns	none	String	This specifies file extensions to be treated as html
ImportDefaultExtension	none	String	If the import module finds a file with an extension that it doesn't know about, it will treat it as the type you enter here, for example .txt or .html
ImportDocumentType	none	String	This is an internal type that gets selected by front ends, for example, Portal which then uses the specified extension type to display an icon.
ImportKeyviewSlaveDirectory	none		This is to specify a central repository for the Keyview DLLs and slave: Importslave.exe should be in the application directory as usual. It will work with or without it. Keyviewslave.exe and the DLLs need to be in the specified directory. Binslave.exe should be in the application directory as usual.
ImportRecursiveDirectoryImporting	False	True/false	When set to true, will specify to recursively import subdirectories.
ImportSlaveTimeout	60	Numeric	This can define the number of seconds the import slave can wait for each document to be imported by Keyview
ImportTempDir	.\	String	This specifies a temporary directory for the importer to use when importing non-standard formats, i.e. documents that are not .txt or .html. It is used to store the temporary file created when the document is converted into an HTML rendered file.
TextImportExtns	none	String	This specifies any file extensions that are to be treated as plain text during import

Content

Parameter	Default	Range	Description
ImportBreaking	False	True/false	Specifies whether or not to break the document into subsections on importing. (Requires the use of combine=1 in the DRE.INI of the engine into which the data is being entered.) ImportBreaking=False HTML files will be broken at any anchors in the page, regardless of the number of words between anchors. Keyview filtered documents will be broken approximately every 3000 words (to the nearest paragraph/sentence/word in that order). PDF documents will be broken on every page regardless of number of words on the page ImportBreaking=on gives the documented behavior. In the case of PDF documents the page number on which the page was broken is also stored.
ImportBreakingMaxParagraphWords	360	Number	Specifies the largest section that can be created when attempting to break the document at paragraph boundaries.
ImportBreakingMinDocWords	600	Number	Specifies the size above which document breaking will be implemented. Smaller documents remain whole.
ImportBreakingMinParagraphWords	160	Number	Specifies the smallest section that can be created when attempting to break the document at paragraph boundaries.
ImportIntelligentTitle Summary	False	True/false	When set to 1, the import module will attempt find unique titles and summaries by comparing the title/summary of each document with the next. So if the first document has the title "News Today", then the import module will set this as its title. If a second document has the same title, then it attempts to find a unique title either from the Headings <H1> <H2> etc.. or failing that, from the content

Parameter	Default	Range	Description
ImportMaxLength	1000000	Number	Any document with size greater than this number of bytes will not be imported. The size of the document will be measured according to the plain text content.
ImportMaxLengthWords	1000000	Number	Any document with more than this number of words will not be imported.
ImportMinLength	0	Number	Any document with size smaller than this number of bytes will not be imported. The size of the document will be measured according to the plain text content.
ImportMinLengthWords	0	Number	Any document with less than this number of words will not be imported.
ImportMinTitleChars	3	Number	Import titles of no less than 3 characters. If this is not found the system will get the title from the content instead.
ImportStoreContent	True	True/false	Specifies whether or not the content of the document is to be stored in the destination DRE.
ImportSummary	False	True/false	Import a field called 'summary' with its content taken from the quick summary generated by the importing module
ImportMaxPlainTextLengthKbs	none		This is the maximum page size criterion. For plain text only (without any tags).
ImportSectionTitles	True	True/false	This allows the titles for each section to be turned off. This is useful when spidering Ecommerce sites that contain an anchor for each product. Disabling the individual section titles means that you can use Autonomy's Distributed Query Handler (DQH) to de-duplicate by title later on (since each review will have the same title)
ImportSummarySize	3	Number	Number of sentences that are to be used in the quick summary generated by the importing module.
ImportTitleStartSkipWords	none		Skip n words before obtaining a title from the page

Appendix A: Import Parameters

Parameter	Default	Range	Description
ImportUnicodeExtns	none	File extension	<p>Enter one or more file extensions to specify which document types should be treated as unicode. Any files of the specified type that are imported will be converted to the character set that you have specified for ImportUnicodeOriginalEncoding.</p> <p>For example:</p> <p>ImportUnicodeExtns=*.html,*.txt</p> <p>ImportUnicodeOriginalEncoding=SHIFTJIS</p> <p>In this example all HTML and Text files that are imported will be converted to SHIFTJIS.</p>
ImportUnicodeOriginalEncoding	none	ASCII GREEK HEBREW ARABIC RUSSIANANSI RUSSIANKOI8 THAI EASTERN EUROPEAN CHINESESIMPLIFIED CHINESETRADITIONAL KOREAN SHIFTJIS UTF8 UCS2	<p>Allows you to specify to which character set you want to convert the unicode document types that you have specified for ImportUnicodeExtns.</p> <p>For example:</p> <p>ImportUnicodeExtns=*.html,*.txt</p> <p>ImportUnicodeOriginalEncoding=SHIFTJIS</p> <p>In this example all HTML and Text files that are imported will be converted to SHIFTJIS.</p>

Parameter	Default	Range	Description
ImportUnicodeAutoDetectLanguage	false	True/false	<p>Allows you to enable language autodetection for unicode documents.</p> <p>If you set <code>ImportUnicodeAutoDetectLanguage</code> to true and a unicode document is imported whose language is recognized, the document is automatically converted to the appropriate character set. If the language that the unicode document contains is not recognized, the document is converted to the character set that you have specified for <code>ImportUnicodeOriginalEncoding</code>.</p> <p>If you set <code>ImportUnicodeAutoDetectLanguage</code> to false, all unicode documents that are imported are converted to the character set that you have specified for <code>ImportUnicodeOriginalEncoding</code>.</p>

Fields

Parameter	Default	Range	Description
FieldName <i>n</i>	none	String	Specifies the name of the Field in the DRE.INI that will be used to store the dynamic field value.
FieldStart <i>n</i>	none	String	This is the string that denotes the start of the value to be stored in the DRE field named in FieldName <i>n</i> .
FieldStop <i>n</i>	none	String	This is the string that denotes the end of the value to be stored in the DRE field named in FieldName <i>n</i> .
FixedFieldName <i>n</i>	none	String	Specifies the name of the Field in the DRE.INI that will be used to store the fixed field value.
FixedFieldValue <i>n</i>	none	String	Specifies the value that will be stored in the DRE field denoted by FixedFieldName <i>n</i> .
ImportChecksum	False	True/false	If set to True, a value is added to the Checksum field in the [Field] section in the DRE.INI. This field value is used to determine whether or not to show a document result in the front-end. This field is typically used with Autonomy Update.
ImportFieldGlueDestination	none	String	This is the destination field to glue ImportFieldGlueSourceCSVs into.
			If you want to glue 2 name fields together you can set FullName=John Smith
			So the ImportFieldGlueDestinationN is to set what the final glued field name is, in this case "FullName" The settings would be: ImportFieldGlueDestination0=FullName

Parameter	Default	Range	Description
ImportFieldGlueSourceCSVs	none	FNAMExxxx / String	<p>This allows you to glue fields together: specify <code>fname<FIELDNAME></code> and / or strings</p> <p>E.G. <code>Field FirstName=John</code> <code>Field SurName=Smith</code></p> <p><code>ImportFieldGlueSourceCSVs0=FNAMExFirstName,</code> <code>"", FNAMExSurName</code> to obtain "John Smith" If you just did: <code>ImportFieldGlueSourceCSVs0=FNAMExFirstName,</code> <code>FNAMExSurName</code> you would get "JohnSmith" or you can add any character/string like: <code>ImportFieldGlueSourceCSVs0=FNAMExFirstName,</code> <code>"", FNAMExSurName," is a star."</code> to get "John Smith is a star" Use "FNAMEx" to denote field value, otherwise it just appends the string as it is.</p> <p>So to recap these two Field Gluing parameters: <code>ImportFieldGlueDestination0=FullName</code> <code>ImportFieldGlueSourceCSVs0=FNAMExFirstName,</code> <code>"", FNAMExSurName</code> to obtain "John Smith"</p>
ImportFieldHTMLConvertChars	False	True/false	In fields, whenever HTML entities are included, they will be replaced with an equivalent character. E.g. <code>&nbsp;</code> will be replaced with a space character.
ImportFieldOPn	none	String	Operations to perform on field (see Field Operations table below) this is a comma separated list of operations.
ImportFieldOpApplyToOn	none	String	The field to apply import operations to
ImportFieldOpParamn	none	String	Parameters for operations specified (see examples in Field Operations table below).

Appendix A: Import Parameters

Parameter	Default	Range	Description
ImportMetaToFields	True	True/false	This specifies whether or not to use HTML meta tags for DRE fields.
ImportRemapField <i>n</i>	none	String	Specifies the name of the field whose value should be used as the value of an alternative field specified by ImportRemapFieldTo <i>n</i> .
ImportRemapFieldTo <i>n</i>	none	String	Specifies the name of the field that will take its value from an alternative field specified by ImportRemapField <i>n</i> .

Field Operations

The operations listed below are only to be used in conjunction with ImportFieldOpParam **n** only.

Parameter	Default	Range	Description
Base64Decode	none		If the text in Base64 encoded this parameter will decode it.
BlankString	none	String	Replaces a specified string with a space EG: Text: What is the meaning of life? 42 ImportFieldOpApplyTO0=Content ImportFieldOP0=BlankString ImportFieldOpParam0="What is the meaning of life?" Content=42
EndAtChars	None	Set of characters, Integer Mode	This specifies which character to end the field at, you can specify which instance of the character you wish to use: nMode=0 – End at leftmost instance of a character in szChars nMode=1 – End at rightmost instance of a character in szChars nMode=2 – End at leftmost char and remove that character nMode=3 – End at rightmost char and remove that character. EG: Text to be imported: <META name="Price" content="£25 plus vat"> ImportFieldOpApplyTO0=Price ImportFieldOP0=EndAt Chars ImportFieldOpParam0="P,1" Price=£25

Appendix A: Import Parameters

Parameter	Default	Range	Description
Escape	none		Escapes text (inserts code for certain characters)
ExtractDigits	none		This extracts digits and inserts them into the field Eg. Price=£24 the field would be populated with 24
ExtractPrice	none		This extracts £ or \$ symbols. So in the previous example to get the £ sign into the field this parameter would also need to be specified. EG: ImportFieldOpApplyTO0=Price ImportFieldOP0=ExtractDigit ImportFieldOpApplyTO1=Price ImportFieldOP1=ExtractPrice Therefore Price should contain £24
GobbleChars	none	Chars	This eliminates multiple duplicated characters, it reduces x instances of the same character to 1. EG: Hellooooo Would show as Hello
GobbleWhiteSpace	none		This eliminates multiple spaces as above this reduces x instances of a space to 1.
HTMLConvertEntities	none		This converts the HTML entities EG: would show as a space
ReplaceLineWithSpace	none		This replaces all carriage returns with a space

Parameter	Default	Range	Description
StartFromLetter	none		If there are preceding numbers in the text they will be ignored and the field will start from the first letter that is found
StartAtChars	none	Set of characters, Integer Mode	<p>This specifies which character to start the field from, you can specify which instance of the character you wish to use:</p> <p>nMode=0 – Start at leftmost instance of a character in szChars</p> <p>nMode=1 – Start at rightmost instance of a character in szChars</p> <p>nMode=2 – Start at leftmost char and remove that character</p> <p>nMode=3 – Start at rightmost char and remove that character.</p> <p>EG:</p> <p>Text to be imported: <META name="Price" content="£25 plus vat"></p> <p>ImportFieldOpApplyT00=Price</p> <p>ImportFieldOP0=StartAtChars</p> <p>ImportFieldOpParam0="£,2"</p> <p>Price=25 plus vat</p> <p>If you use StartAtChars in conjunction with EndAtChars you could narrow the field down even further:</p> <p>ImportFieldOpApplyT00=Price</p> <p>ImportFieldOP0=StartAtChars,EndAtChars</p> <p>ImportFieldOpParam0="£,2,p,3"</p> <p>Price=25</p>

Appendix A: Import Parameters

Parameter	Default	Range	Description
StripChars	none	Chars	This removes specified characters EG: ImportFieldOpApplyTO0=Author ImportFieldOP0=StripChars ImportFieldOpParam0=" & ! #" Will remove all occurrences of & ! #
StripHTML	none		This will remove all remaining HTML Tags
StripNumbers	none		This will remove all numbers
TailWhiteSpace	none		This will remove any extra spaces from the end of the field
TerminateAtInvalid	none		This truncates the string if the character found is not a letter/space/number. This will terminate if an invalid character is found.
TerminateAtLine	none		This terminates at a carriage return
TerminateAtSpace	none		This terminates at a space.
TopNTailWhiteSpace	none		This will remove any extra spaces from the beginning and the end of the field
UnEscape	none		This unescapes text where it is already escaped.

DRE

Parameter	Default	Range	Description
Database	Database specified during install	String	Specifies the name of the database into which documents will be indexed.

Date

Parameter	Default	Range	Description
ImportExtractDateFrom	0	Number. Can be either: 0 (Nothing) 1 (Current time - now) 2 (last accessed date) 4 (Created time) 8 (last modified date) 16 (from DRE field) 32 (from content) 64 (from filename)	Specifies what date is to be extracted from the document.
ImportExtractDateFromField	none	String	Specifies the name of a date field in the DRE whose value is to be extracted.

Appendix A: Import Parameters

Parameter	Default	Range	Description
ImportExtractDateFor matCSVs	none	String	<p>These specify the format in which dates are to be extracted. Comma separated date formats. These format specifiers are used whenever dates can be specified in the DDMYY format style strings.</p> <p>Format specifiers:</p> <p>YY – Year (2 digit) , e.g. 99 or 00 or 01 etc.</p> <p>YYYY - Year (4 digit), e.g. 1999 or 2000, 2001 etc</p> <p>LONGMONTH – January, March, August etc.</p> <p>SHORTMONTH -.Jan, Mar, Aug etc.</p> <p>MM – Month (2 digit) 01, 10, 12 etc</p> <p>M+ - Month (1/2 digit) 1,2,3,10 etc.</p> <p>DD – Day (2 digit) 01, 02, 03, 12, 23 etc.</p> <p>D+ - Day (1/2 digit) 1, 2, 12, 13, 31 etc</p> <p>LONGDAY (2 digit with postfix) 1st 2nd etc.</p> <p>HH – Hour (2 digit) 01, 12, 13 etc.</p> <p>H= - Hour (1/2 digit)</p> <p>NN – Minute (2 digit)</p> <p>N+ - Minute (1/2 digit)</p> <p>SS – Second (2-digit)</p> <p>S+ - Second (1/2 digit)</p> <p>ZZZ – Time Zone (GMT, EST, PST, etc.)</p> <p>Examples:</p> <p>// Dates with 1/2 digit days</p> <p>DateFormats=D+/SHORTMONTH YYYY, DDMYY</p> <p>//Quoted string to allow spaces and commas etc within the format</p> <p>DateFormats="D+SHORTMONTH YYYY", "Date: D+LONGMONTH, YYYY"</p> <p>//Directory style dates</p> <p>DateFormats=D+/M+/YY, MM/DD/YYYY etc.</p>

Parameter	Default	Range	Description
ImportExtractDateToField	DREDATE	String	Specifies the name of a date field in the DRE whose value will be taken from ImportExtractDateFromField
ImportExtractDateToFormat	none	String	Specifies the format of the dates extracted once in the DRE. NB: If ImportExtractDateToField=DREDATE, then ImportExtractDateToFormat=YYYY/MM/DD. OR Number of seconds since 1970 OR NOW These formats MUST all be specified in uppercase EG: YYYY/MM/DD this is how the DRE understands it. Seconds since 1970 is the default so leave the setting out.
ImportExtractLength	False	True/false	When set to True, will allow the extraction of the file length.
ImportExtractLengthToField	FILELENGTH	String	When ImportExtractLength=1, the file length will be extracted into the DRE field specified by this parameter. E.g. ImportExtractLengthToField=FileLength

Page Layout

Parameter	Default	Range	Description
HTMLImportEndDefCSVs	none	String	Comma separated list of strings that are used to mark the end of text in an HTML document to be indexed into the DRE.
HTMLImportStartDefCSVs	none	String	Comma separated list of strings that are used to mark the beginning of text in an HTML document to be indexed into the DRE.
HtmlImportTurnToSpaceEndDefCSVs			This specifies the end string in an HTML document to remove the content between the stated StartDefCSVs and EndDefCSVs from an imported file

Appendix A: Import Parameters

Parameter	Default	Range	Description
HtmlImportTurnToSpaceStartDefCSVs			<p>This specifies the starting string in an HTML document to remove the content between the stated StartDefCSVs and EndDefCSVs from an imported file</p> <p>For Example:</p> <pre><html> <head> </head> <body></pre> <p>Nice content</p> <pre><table> START Nasty content END </table> More nice content </body> </html></pre> <p>Then setting</p> <pre>HtmlImportTurnToSpaceStartDefCSVs =<table> START HtmlImportTurnToSpaceEndDefCSVs=END </table> HtmlImportTurnToSpaceStripTags=true</pre> <p>These settings would remove the nasty content, START and END from the imported file (actually turns it to space, but the DRE ignores it).</p> <p>To do this you must have HtmlImportTurnToSpaceStripTags set to true.</p>

Parameter	Default	Range	Description
HtmlImportTurnToSpaceStripTags	none	True/false	Specifies that the content between the start and end tags is to be removed and a space will replace it.
ImportEndDefCSVs	none	String	Comma separated list of strings that are used to mark the end of the text in a document to be indexed into the DRE.
ImportPageBreakDefs	none	String	Specifies a string that is used to mark a document break. This is used for splitting up documents into segments with each segment being contained in one idx format. When the idx file have been indexed into the DRE, the individual segments are joined back together to produce the original document.

Path

Parameter	Default	Range	Description
ImportPathReplaceString	none	String	Specifies a string that is to replace the string preceding the nth slash in a document reference. ImportPathReplaceUpToSlash specifies the nth slash.
ImportPathReplaceUpToSlash	-1	Number	Specifies the nth slash where the preceding string is replaced with the string specified in ImportPathReplaceString. This is used so that a single portion of many strings which contain different substrings can be replaced as opposed to just one particular word. E.g. In the case where the files in the queue are C:\a\b\hello.c:\a\chello and c:\b\a\hello ImportPathReplaceUpToSlash=3 would replace the portion if the strings up to the third back-slash in each string.
ImportRefReplaceCSVs	none	String	Specifies the string that is to be replaced.

Appendix A: Import Parameters

Parameter	Default	Range	Description
ImportRefReplaceWithCSVs	none	String	Specifies the string that is to replace the original reference of the file.
ImportRefTruncateAfter	- 1	Number	Specifies that the reference must be truncated after the nth occurrence of the string specified in ImportRefTruncateString.
ImportRefTruncateString	none	String	Specifies the string that is used to truncate the reference after the nth occurrence.
ImportRegisterExtnCSVsN=	none	String	N= represents an infinite number (starting with 0, MUST BE SEQUENTIAL). This specifies the files with certain extensions that will be imported. E.g. importRegisterExtnCSVs0=HTML, XML
ImportRegisterExecutableN=	none	String	N= represents an infinite number (starting with 0 MUST BE SEQUENTIAL). This specifies the import slave that will import the corresponding files (see importRegisterExtnCSVs). E.G. ImportRegisterExecutable0=testslave In this example Testslave will import HTML and XML files.
ImportStartDefCSVs	none	String	Comma separated list of strings that are used to mark the beginning of the text in a document to be indexed into the DRE.
ImportStartSkipSentences	0	Number	Specifies the number of sentences to skip from the beginning of a document when importing a document.
ImportStartSkipWords	0	Number	Specifies the number of words to skip from the beginning of a document when importing a document.

Reference Replace is used in the following way:

When the directory is imported, the reference to that file becomes something like, C:\lnetpub\wwwroot\filename. Now if you want to store the actual reference for this file so that you can access its URL, then you need to replace the C:\lnetpub\wwwroot with http://hostname/. The following lines will perform this in the configuration file:

ImportRefReplaceCSVs=C:\lnetpub\wwwroot

ImportRefReplaceWithCSVs=http://hostname/

The order in which the strings are replaced goes from left to right so be aware of the order you put your strings in.

E.g.

```
ImportRefReplaceCSVs=C:\Inetpub\wwwroot\,
ImportRefReplaceWithCSVs=http://hostname/,
```

will replace the string,

```
C:\Inetpub\wwwroot\directoryname\myfile with
http://hostname/directoryname/myfile
```

But `ImportRefReplaceCSVs=\\C:\Inetpub\wwwroot\`

```
ImportRefReplaceWithCSVs=/,http://hostname/
```

Will replace the string,

```
C:\Inetpub\wwwroot\directoryname\myfile with
C:/Inetpub/wwwroot/directoryname/myfile
```

The `Import Reference Replace` function can be applied more than once and this is used by setting the value of the `ReferenceReplaceString` and `ReferenceRplaceWith` parameters to be a list of comma separated paths.

E.g. `ImportRefReplaceCSVs=C:\Inetpub\wwwroot\, D:\Inetpub\wwwroot\`

```
ImportRefReplaceWithCSVs=http://193.128.174.23/, http://193.128.174.38/
```

This will replace the string `C:\Inetpub\wwwroot\` with `http://193.128.174.23/` and `D:\Inetpub\wwwroot\` with `http://193.128.174.38/`

whenever they appear in the queue of files to be processed. You must therefore, also ensure that the number of paths specified in `ImportRefReplaceCSVs` is equal to the number of paths in `ImportRefReplaceWithCSVs`.

Reference replacement is recursive, which means that if you want to replace the string `C:\Inetpub\wwwroot\data\data1` with `http://127.0.0.1/data/data1`.

Then do `ImportRefReplaceCSVs=C:\Inetpub\wwwroot,`

```
ImportRefReplaceWithCSVs=http://127.0.0.1/
```

ImportPathReplaceUpToSlash and ImportPathReplaceString are both performed before ImportRefReplaceCSVs and ImportRefReplaceWithCSVs.

Import Rendering

Parameter	Default	Range	Description
ImportRenderedHTMLExtensions	none	String	Specifies the extension of files on which to perform HTML rendering, this should be *.doc.htm
ImportRenderedHTMLFieldName	none	String	For any non HTML file that gets imported, the HTML file that comes out of Keyview/PDF gets moved into the ImportRenderedHTMLFieldName field.
ImportRenderereHTMLMoveToDir	none	String	For any non-HTML file that gets imported, the HTML file that comes out of Keyview/PDF gets moved into the ImportRenderedHTML MoveToDir directory.
ImportRenderedHtmlPathReplaceString	none	String	Specifies the replacement string of the string immediately preceding the nth '\'.
ImportRenderedHTMLPahtReplaceUpToSlash	-1	Number	Specifies a string that is to be replaced up to a certain '\'. Used to set the right reference for the ImportRenderedHTMLFieldName field.
ImportRenderedHTMLRefReplaceCSVs	none	String	Specifies the string that is to be replaced. Used to set the right reference for the ImportRenderedHtmlFieldName field.
ImportRenderedHTMLRefReplaceWithCSVs	none	String	Specifies the string that is to replace the original reference of the ImportRenderedHtmlFieldName field. Used to set the right reference for the ImportRenderedHtmlFieldName field.

The parameters needed to for Import Rendering are:

ImportRenderedHTMLMoveToDir
 ImportRenderedHTMLFilenameExtensions
 ImportRenderedHTMLFieldName

You can also perform any reference replacement necessary to get the right path. These settings are equivalent to the set used for the DRE document reference.

ImportRenderedHTMLPathReplaceUpToSlash
 ImportRenderedHTMLPathReplaceString
 ImportRenderedHTMLPathReplaceCSVs
 ImportRenderedHTMLRefReplaceWithCSVs

The KeyView and PDF slaves temporarily convert all the various formats (i.e. word, PDF, Excel, PPT etc.) into HTML. This rendered HTML is then processed to create data in IDX ASCII format. If you wish, you can store the temporary HTML data on a repository on the web server.

Parsing

Parameter	Default	Range	Description
ImportStripLinks	True	True/false	When set to True, will remove all content that is a hyper text link.
ImportStripStrings	none	String	Comma separated list of strings that, when matched in a document, are to be stripped from the idx format. NOTE: DO NOT enter punctuation in the string.
ImportStripTagCSVs	none	String	Strips the content between begin and end tags. E.g. ImportStripTagCSVs=B Will strip all content between and .

Delimited

Parameter	Default	Range	Description
ImportDelimitedDocStart	none	String	This specifies where the beginning of a record is when you are importing a delimited file
ImportDelimitedDocEnd	none	String	This specifies where the end of a record is when you are importing a delimited file.
ImportDelimitedEndn	none	String	This specifies the end string of data to be inserted into the idx field. The name of this field is specified by ImportDelimitedFieldn.
ImportDelimitedExtns	none	String	This specifies the extension of the files which are to be treated as delimited files. E.g. .CSV files

Appendix A: Import Parameters

Parameter	Default	Range	Description
ImportDelimitedFillIn RefFromHTML		True/false	When doing delimited importing you can normally specify to get the reference from the delimited text you are importing. If you don't want to do that and just use the usual reference you can set this to true. Useful when fetching HTML pages with delimited data in it. The reference should still be the URL of the site, hence "Fill in Reference from HTML", which is set by the spider.
ImportDelimitedField <i>n</i>	none	String	There is a field where data from a delimited file is stored. This specifies the name of that field. ImportDelimitedStart <i>n</i> and ImportDelimitedEnd <i>n</i> specify the data to include in that field.
ImportDelimitedSkip Chars <i>n</i>	0	Number	When the idx file is produced a number of characters will be skipped at the beginning of a document. This specifies how many characters to skip.
ImportDelimitedStart <i>n</i>	none	String	This specifies the start string of data to be inserted into the idx field. This idx field will be denoted by ImportDelimitedField <i>n</i> .

XML

The order to specify the XML import parameters are as follows:

```
ImportXMLExtens=".XML"
ImportXMLField0=DRREFERENCE
ImportXMLSearchCSVTags0=description,document
ImportXMLEntryOnly0=True
ImportXMLStripTags0=True
```

Parameter	Default	Range	Description
ImportXMLAttribute ⁿ			<p>This specifies the XML attribute that is to be imported.</p> <p><PRODUCT ID="9999"> Text1 More Text </PRODUCT></p> <p>here ID is an attribute of the tag PRODUCT.</p> <p>So if the following is specified:</p> <p>ImportXMLAttribute0=ID</p> <p>Will then import the ID attribute of the Product tag.</p> <p>TODO – how where when etc...</p>
ImportXMLEntryOnly ⁿ	False	True/false	<p>Only imports the content between the searched tags at that level.</p> <p>For example if an XML file contained the following strings:</p> <pre><document> <description> Text1 <document> Text2 <description2> more text </description2> </document> </description> </document></pre> <p>and</p> <p>ImportXMLSearchCSVTags0=description,document</p> <p>ImportXMLEntryOnly0=True, then the content imported = "Text2"</p> <p>If ImportXMLEntryOnly0=False, then the content imported = "Text2 more text"</p>

Appendix A: Import Parameters

Parameter	Default	Range	Description
ImportXMLExtns	none	String	A comma separated list of all XML file extensions. E.g. ImportXMLExtns=*.xml
ImportXMLField <i>n</i>	none	String	Will specify the field in the DRE where the content taken from the XML File will be stored. Content as defined by the parameter ImportXMLSearchCSVTags <i>n</i> . For example if an XML file contained the following strings: <head> my example </head> and ImportXMLSearchCSVTags0=head and ImportXMLField0=DREREFERENCE Then the DRE field DREREFERENCE will contain in it text "my example"

Parameter	Default	Range	Description
ImportXMLSearchCSVTagsn	none	String	<p>Searches for the appropriate XML tag(s). This can be a hierarchy level of search tags. It uses the tags specified to navigate to the level you wish to retrieve data from then within the lowest level tag it will get all the data between the <> and </>, so for example:</p> <p>Document contains:</p> <pre><document> <description> Text1 </document> Text2 </document> </description> </document></pre> <p>So, if you wanted text from <document> that is located within <description>, then you will need to specify as follows:</p> <p>ImportXMLSearchCSVTags0=description,document</p> <p>or</p> <p>ImportXMLSearchCSVTags0=document,description,document (both gives same results)</p> <p>but if you just put</p> <p>ImportXMLSearchCSVTags0=document</p> <p>Everything within the first instance of <description> will be retrieved. ie "Text1"</p> <p>Basically, the CSVTags allow to user to define the exact search criteria for a specific tag that a user wants.</p>

Appendix A: Import Parameters

Parameter	Default	Range	Description
ImportXMLStripTags <i>n</i>	False	True/false	<p>If set to true, will strip the remaining XML tags (if any exist) between the specified XML tags upon import of the file. For example</p> <pre><document> <description> Text1 <document> Text2 </description2> more text </description2> </document> </description> </document></pre> <p>and</p> <p>ImportXMLEntryOnly0=False</p> <p>ImportXMLField0=DRREFERENCE</p> <p>ImportXMLSearchCSVTags0=description,document</p> <p>Then the field would be as follows:</p> <p>DRREFERENCE= Text2 more text</p> <p>NOT</p> <p>DRREFERENCE=Text2 <description2> more text </description2></p>

The XML import section is quite tricky to get data required, please find below some more examples that may help.

1. If the tag is defined as the following `<productarticle ui="1234" languagecode="EN">` you define the tag as follows:

`ImportXMLSearchCSVTagsn=productarticle`

2. If the tag with in the document is unique there is no need to worry about nesting, however, if the same tag appears again in the document, you can list the nesting levels to get down to the required tag, I.E.:

In the example below there are two tags labeled `<charts>`, the data to be retrieved is the content from the last `<charts>` tag, so one would need to specify the following:

`ImportXMLSearchCSVTag0=media,picture,charts,datasheets,charts`

```
<media>
  <picture filename="een.jpg" />
  <charts>
    <chart filename="chart.xxx" />
    <chart filename="chart.yyy" />
  </charts>
  <datasheets>
    <datasheet filename="xxx.pdf" />
    <datasheet filename="yyy.pdf" />
  </datasheets>
  <charts>
    <appnote filename="zzz.pdf" />
    <appnote filename="aaa.pdf" />
  </charts>
</media>
```

3. It is **MANDATORY** to populate the DRREFERENCE with content, from one of the tags, other fields to consider:

DRREFERENCE – Mandatory

DRETITLE – Optional

DRECONTENT - Optional but makes sense to have it

Logging

Parameter	Default	Range	Description
ImportLogFile	Queryh.log	String	Specifies the name of the file used to record the import process.
ImportSetMaxLogFile	Size 1000000 0		This specifies the max size the log file is allowed to reach before it is deleted/archived and recreated. This is for the import log only set by ImportLogFile.
ImportLogging	False		This turns import logging on and off!
ImportLogLevel	ERRORS	ERRORS WARNING S NONE FULL	Specifies the type of login that is to be used.
ImportLogFileAppend	False	True/false	Boolean value specifying whether or not details are to be appended to an existing import log file.

Binary Content

Parameter	Default	Range	Description
ImportFilterBinaryContent	False	True/false	Specifies that binary content is to be processed.
ImportMaxAverageWordLength	25	Number	Binary words with more than this number of characters will not be imported.
ImportMaxAllowedASCII CODE	175	Number	The maximum ascii value allowed in characters in the binary content. Anything higher is treated as binary and removed.
ImportMaxBinaryCharactersPerHundred	3	Number	Maximum number of binary characters tolerated per 100 characters. Anything binary exceeding this is to be ignored.
ImportMinAllowedASCII CODE	0	Number	The minimum ascii value allowed in characters in the binary content. Anything lower is treated as binary and removed.

Email Parsing

Parameter	Default	Range	Description
ImportAttachments	False	True/false	Specifies that email attachments are to be imported.
ImportSaveAttachments	False	True/false	Specifies that email attachments are to be saved to the directory specified by ImportSaveAttachmentDir.
ImportSaveAttachmentDir	.\	String	Specifies the directory path where email attachments are to be saved.

Import Anchors

Parameter	Default	Range	Description
ImportAddHtmlAnchorNames	True	True/false	Adds the name of the anchor to the end of the title for each title in the HTML page
ImportMaxAnchors	None		This specifies how many anchors to return. It will process up to this number of anchors in one HTML page. After this max number it will stop. It prevents from processing pages that are too long.

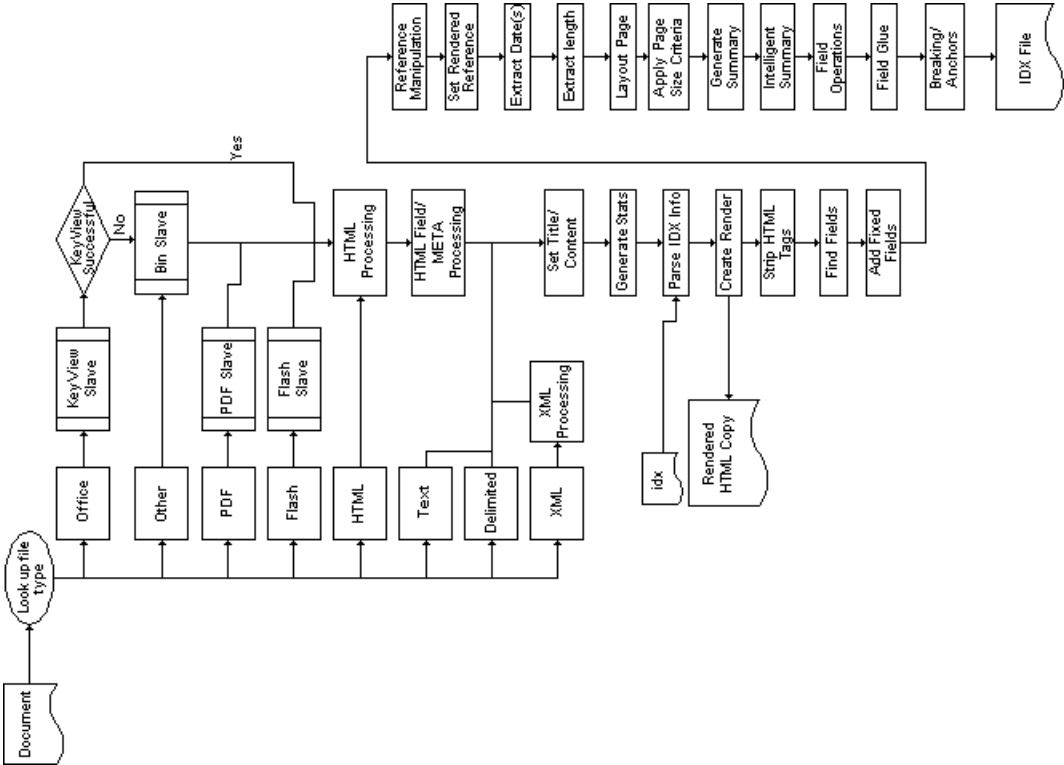
The import module will automatically perform the breaking of HTML pages on anchors. It keeps the reference to the actual anchor, so that when clicking on a result, you go straight to that anchor section rather than the top of the page.

If you want to use this feature, you just need to make sure that ImportBreaking=on and you need to set the breaking parameters to be sensible according to the size of the data in each anchor. It will only break on anchors if the breaking parameters are set so that it is possible.

When breaking on anchors you are advised to use the ImportIntelligentTitleSummary=1 so that titles and summaries are of the individual anchor sections rather than the page in general. Otherwise all summaries and most titles will be all the same.

Import Process Flow

The sequence in which the import process executes the parameters specified is represented in the following diagram:



Appendix B: DRE Security Modules

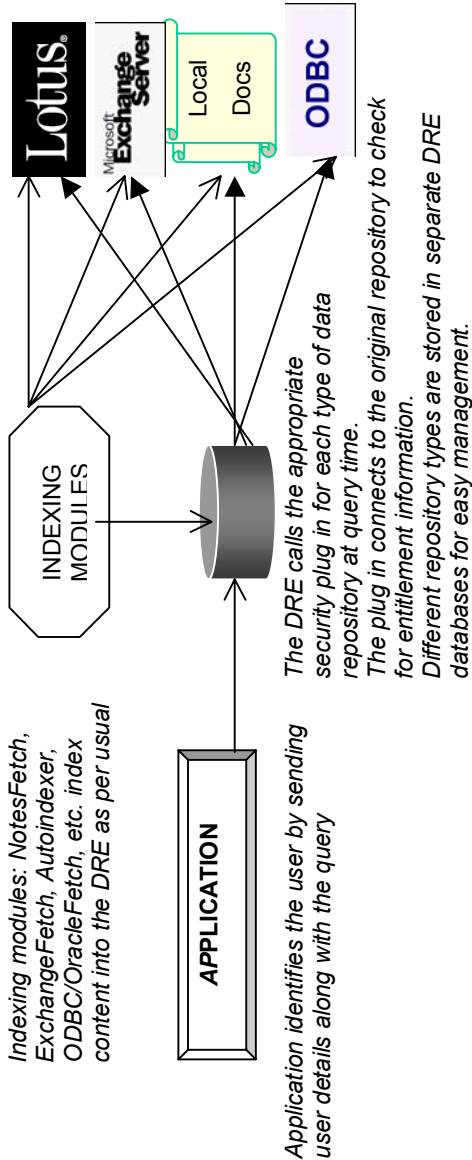
Autonomy Systems Ltd is committed to providing integration with existing third party repositories in order to enforce security restrictions. This is known as enforcing the entitlement that a user has to view one or more documents. The Autonomy Dynamic Reasoning Engine (DRE) security plug in modules ensure that only documents which the user is entitled to see are sent back to the requesting application. This, of course, requires the user to identify himself to the DRE.

You should not confuse entitlement with authentication. Authentication only checks whether the user is allowed to enter the system whereas entitlement ensures that documents can only be shown if the user is allowed to see them.

There are 2 modes of security that Autonomy have implemented: Mapped and Non Mapped.

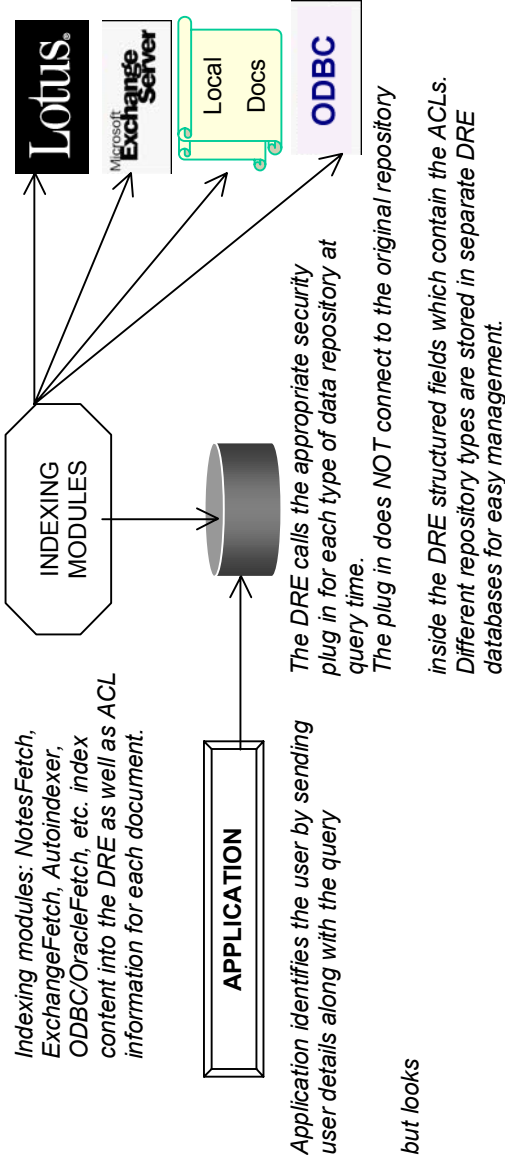
NON MAPPED SECURITY

The security entitlement check is done in real time by the DRE. The DRE needs direct connection to the data repository in order to use any necessary APIs to obtain access control information.



MAPPED SECURITY

The Access Control List (ACL) of the documents in the repository is mapped into a structured field in the DRE. These structured fields are encrypted to ensure confidentiality. The indexing module retrieves the ACL for each document at the time of indexing and stores them in the DRE. The DRE does not need any connection with the repository itself.



The table below outlines the unique features of each type of security implementation together with its advantages and disadvantages.

	NON MAPPED	MAPPED
ADVANTAGES	100% in sync with the entitlement information in the repository.	Much faster response time. From an infrastructure point of view, the DRE does not need direct connection with the content repository.
DRAWBACKS	<p>Response can be slow as entitlement lookup is done for each result sent back by the DRE. Entitlement lookup is done by connecting to the original repository which increases the overhead. Caching can be used to alleviate this.</p> <p>Also the DRE needs to process as many documents as necessary to find relevant results to send back. The worst case scenario would be if the user is entitled to see none of the documents, the DRE would have to process every single relevant document in the database to finally say that there are no results.</p>	<p>Indexing is slower as it needs to retrieve the ACL for each document in order to store them in the DRE.</p> <p>There will be a slight lag between a security setting being changed in the content repository and it being indexed into the DRE, and hence available to the application.</p>
USAGE	Good for environments where the security entitlement for documents changes frequently, and where the average user is, on average, allowed to see more than 80% of the data.	Good for environments where the security entitlement does not change too often, and where indexing speed is not necessarily an issue.
REQUIRED MODULES	Non mapped version of the security plug in DLLs for the repositories used.	Mapped version of the security plug in DLL and a version of the indexing module that retrieves ACLs.

DRE and Security DLLs

Autonomy Systems Ltd currently supplies the following security plug in DLLs:

	UNMAPPED	MAPPED
Lotus Notes	Y	Y
ODBC/ORACLE	Y	N
NT	Y	Y
Exchange	Y	Y
UNIX	N	Coming soon
NDS	N	Coming soon

The DLLs act as plugins for the DRE itself. The DRE uses them when instructed to do so in the configuration file.

The DRE goes through the following process:

1. On startup it loads the DLLs
2. When the DRE receives a query, it looks for the best matching results.
3. For each result, it calls the security DLL to find out if the requesting user has permission to see that document. If YES then the document is shown, if NO then it isn't. The DRE makes sure that if N results are requested, the results shown are N documents that the user is allowed to access.

The input parameters for the security DLL are:

- Username, Password, Domain, Group (passed through from the front end application)
- Document Reference (can be configured to be any structured field in the DRE)

To communicate with the DRE directly, you can pass the following arguments in the HTTP Query:

```
user=<User Name>
pass=<Password>
domain=<Domain Name>
group= <Group Name>
```


For example:

`/qmethod=q&qquerytext=accounts&user=John&pass=111&domain=Global&group=Admin,Developers`

The mandatory variable/value pairs required vary according to the type of security implemented. For each security type these mandatory values are:

SECURITY TYPE	UNMAPPED	MAPPED
Notes	<ul style="list-style-type: none">• user: Notes user name.• group: Comma separated list of groups that the user belongs to.• domain: Notes domain name.	<ul style="list-style-type: none">• user: Notes user name• group: Comma separated list of groups that the user belongs to.
ODBC	Dependent on the stored function or procedure.	N/A
NT	<ul style="list-style-type: none">• user: NT username.• pass: NT password.• domain: NT domain name.	<ul style="list-style-type: none">• user: NT username.• domain: NT domain name.• group: Comma separated list of groups that the user belongs to.
Exchange	<ul style="list-style-type: none">• user: Exchange user profile name• domain: Exchange server domain name.	N/A
Unix	N/A	N/A

N.B. To find out how to configure an Autonomy front end such as Update, Portal In A Box, etc., to pass through these parameters please refer to the appropriate front end manual.

DRE Generic Security Configuration

In order to successfully set up the DRE for security, you must use **DRE version 3.1.1 or later**.

The configuration of the DRE.INI file is done in the same way for all types of security, with some different specific values for certain settings. You need to add a security entry for the DRE database(s) in DRE.INI. For example:

[Databases]
NUMDBS=1
O=**ARCHIVE**

[**ARCHIVE**]
Security=**MySecuritySection**

You need to add a Security= entry under the database name. The value of this entry is the name of the security section, in this example **[MySecuritySection]**.

[MySecuritySection]	
Library=	Logging=
Type=	DebugDecrypt=
ReferenceField=	CacheMaxSize=
HideField=	CacheExpiryMinutes=

ENTRY	DESCRIPTION	DEFAULT
Library	The name of the DLL file to be used. This should be the name of the DLL supplied by Autonomy Systems Ltd.	Not specified. No security enabled.
Type	The type of security you are using, the possible values are: AUTONOMY_SECURITY_NOTES AUTONOMY_SECURITY_NOTES_MAPPED AUTONOMY_SECURITY_ODBC AUTONOMY_SECURITY_ORACLE AUTONOMY_SECURITY_NT AUTONOMY_SECURITY_NT_MAPPED AUTONOMY_SECURITY_EXCHANGE AUTONOMY_SECURITY_EXCHANGE_MAPPED AUTONOMY_SECURITY_UNIX_MAPPED	Not specified. No security enabled.
ReferenceField	The field used to identify the document against which the security entitlement must be matched. By default this is the DRREFERENCE of a document. Alternative fields can be specified in this entry. For example, in the mapped version of the security modules the value of this entry will typically be: ReferenceField=AUTONOMYMETADATA	AUTONOMYMETAD ATA
HideField	If the value of ReferenceField is set to something other than the DRREFERENCE, you can set HideField=1 to avoid the field being shown within the results that the DRE sends back. This is primarily to be used when implementing mapped security, so that the encrypted ACLs are not shown as part of results set.	FALSE = 0 = NO
Logging	For troubleshooting purposes, you can set Logging=1 to obtain logging information into a file called security.log. Entries in this file will give you an idea of exactly what the DLL is doing. It is important to turn this setting off once the system is configured properly to avoid it slowing down.	FALSE = 0 = NO

ENTRY	DESCRIPTION	DEFAULT
DebugDecrypt	Provides a finer grain of Logging. If DebugDecrypt=1 it will print out the ACL information that the DLL is matching against. This is only recommended in extreme cases as it is a security risk to have ACL information printed out in plain text in a file. This setting comes into effect only when Logging=1. This setting is available only for the mapped version of the security plug ins.	FALSE = 0 = NO
CacheMaxSize CacheExpiryMinutes	The unmapped version of the DLLs can cache security information as it becomes available so that the next time the same request is made the results are returned faster. CacheMaxSize: Maximum Cache Size. Default: 1000 CacheExpiryMinutes: Time in minutes after which each cached entry should expire. Default: 0 (no expiry)	1000 0

In addition to these generic settings, certain types of security implementation will require further configuration entries.

Lotus Notes Security – Unmapped



Implementation

The Notes Security DLL follows these steps to ascertain whether a user is allowed to see a document:

1. From the document reference it extracts the Notes database name and the Notes ID of the document.
2. Open the Notes database with the path <Server Name>\<Domain Name>!<Database Name> e.g. Technical\Autonomy!\api.nsf
3. Extracts the Database ACL and checks if the user has permission to access the database.
4. If yes, then it carries on to check the Readers Field from the document Note. If the user matches any of them, access is granted.

Set up

To implement Notes security in the DRE please follow these steps:

1. Make sure you have DRE version 3.1.1 or higher.
2. Make sure you have a Notes Client installed on the same machine as the DRE. The Notes Client must be configured to use a Notes ID which has administrator rights, such as viewing of ACL information with access to the database.
3. Copy the Notes security DLL supplied by Autonomy Systems Limited in the DRE directory. This DLL can have any name, e.g. call notes_security.dll
4. Copy the nextpwd.dll file into the DRE directory.
5. Make sure that the notes installation directory is in the PATH variable.
6. Create a file called "notesfetch.cfg" in the DRE directory with an entry like

```
[Passwords]
C:\notes\data\user.id=autonomy
```

This entry is a <Full Path to User ID File>=<Password for that User ID> pair value. This should be the administrative access account that the DRE uses to access the ACL from the Notes database, in order to perform the security check.

Appendix B: DRE Security Modules

7. Edit the notes.ini file typically stored in the windows directory and add the following line:

EXTMGR_ADDINS=EXTPWD,NEXTPWD.DLL

8. Configure the DRE.INI. For example:

[Databases]
NUMDBS=1
0=**ARCHIVE**

[**ARCHIVE**]
Security=**Notesf**

[**Notes**]
Library=notes_security.dll
Type=AUTONOMY_SECURITY_NOTES

Lotus Notes Document References

The Notes security DLL has been designed to process references created by the NotesFetch or by spidering the Domino front end.

The values of DRREFERENCE for Notes must have one of the following forms:

- A. Document reference set to be accessed via Notes Domino Web Server (with external server name known):
<http://myserver.mycompany.com/dbaset.nsf/By+Author/54e2dd8497363ced802567cf005cbf56?OpenDocument>
- B. Document reference set to be accessed via Notes Domino Web Server (with internal server name known):
<http://myserver/dbaset.nsf/By+Author/54e2dd8497363ced802567cf005cbf56?OpenDocument>
- C. Document reference set to be accessed via Notes Client:

`http://myserver /scripts/delivernotes.exe?replica=802567cf:005cb79b&view=OFc317180a:11767f07-ON85256499:006b15a3&nsfnote=OF54e2dd84:97363ced-ON802567cf:005cbf56&server=myserver`

If the DRE is dealing with references of type A above then you will need to add the following settings to the security configuration section:

```
ServerNamesFrom=myserver.mycompany.com
ServerNamesTo=myserver
```

This forces a string replacement for the server name, so that the Notes security DLL knows that the Notes server corresponding to "myserver.mycompany.com" is "myserver". Notes security needs to know the name of the Notes server rather than the full web server name.

In addition to this you can also set general reference replacements by using:

```
RefReplaceStringsFrom=Document+Management
RefReplaceStringsTo="Document Management"
```

These settings are particularly useful when dealing with database names with a space in between. The URL reference uses a plus instead of the space, however the security DLL needs to have the original name to connect to the database. With the conversion set above, you can ensure that the security DLL will connect with the correct string.

LDAP type usernames

When using LDAP compliant usernames, you must make sure that it is of the form that Notes understands. For example, Notes can have users of the forms:

“Jo Bloggs”

“CN=Jo Bloggs/O=mycompany”

“CN=Jo Bloggs/OU=marketing/O=mycompany”

The exact string needs to be formed for the security comparison to take place successfully.

The security DLL checks for both cases <USERNAME> and CN=<USERNAME> to allow for both cases. Where additional Organisation Unit (OU) and/or Organisation (O) name are supplied, you must make sure that these get sent through from whichever front end module you are using.

So for example in Portal In A Box, you can set the username for the Notes Database security to be “Jo Bloggs/OU=marketing/O=mycompany”.

Troubleshooting

If you set Logging=1 you might encounter the following error message(s) in the security.log file:

NOTES ERROR MESSAGE	REASON
Unable to open database: Access Denied	You must make sure that the Notes User ID that the Notes Client uses when run from the server where the DRE is running on is that of an administrator account. The security DLL uses this User ID to connect to Notes.

Lotus Notes Security – Mapped



Implementation

This security setup assumes the use of the NotesFetch to extract and index Notes content into a DRE. The implementation components would be as follows:

- The *DRE* contains indexed documents in which each document will have an encrypted structured field containing the ACL information for that document (this requires DRE version 3.1.1 or above).
- The *DRE Notes Security DLL* which performs the ACL comparison operation.
- The *NotesFetch* which indexes Notes documents into the DRE including the ACL information of those documents (this requires NotesFetch version 2.1.1 or above).

Component Features and Configuration

First of all make sure you already have the following in place:

- DRE version 3.1.1 or above
- NotesFetch version 2.1.1 or above

Dynamic Reasoning Engine

Features:

- The DRE holds the ACL information in a structured field which is both hidden and encrypted. Hidden means that it is only used internally, it is not indexed in any way and it is never shown in result sets.
- The DRE receives as parameters, the Notes user name AND the group which the user belongs to. Password and Domain information for the user is NOT required.

Configuration:

To configure the DRE you specify the security details per database by setting *Type=AUTONOMY_SECURITY_NT_MAPPED* and a *ReferenceField* which is the field where the ACL information is to be stored. This can be changed to anything you want, but you'll need to make sure that the name is reflected in the other components as well. By default it is called *AUTONOMYMETADATA*. *HideField=1* is set for security purposes, so that the field containing the ACL information is never returned as part of a result in a search.

Appendix B: DRE Security Modules

For example:

```
[Databases]
NUMDBS=1
0=MyDB

[MyDB]
Security=NotesSec

[NotesSec]
Library=notes_mapped_security.dll
Type= AUTONOMY_SECURITY_NOTES_MAPPED
ReferenceField=AUTONOMYMETADATA
HideField=1

[Fields]
NumFields=3
0=DRREFERENCE
1=INDEX10.DRETITLE
2=AUTONOMYMETADATA
```

In the [Fields] section you must have a field listed which is the same as the value in *ReferenceField*.

Parameters:

The parameters required to be sent to the DRE are:

user=<username>&group=<group>

NotesFetch

Features:

- The NotesFetch indexes the initial ACL information at the same time the Notes content is indexed into the DRE. At any subsequent cycle the NotesFetch will detect changes in the ACL, as well as in the content, and update them in the DRE.
- The NotesFetch must run with the Notes ID of a user who has administrative privileges, this is because it needs to have access to the ACL information stored within Notes.

Configuration:

Per Notes Job you can configure:

NotesACLMapping=1
NotesSecurityField=AUTONOMYMETADATA
ACLReadersFieldCSVs=\$Readers

CONFIG ENTRY	DESCRIPTION	DEFAULT
NotesACLMapping	Turns the security on, it tells NotesFetch to index the ACLs.	0
NotesSecurityField	Indicates the structured field in the DRE where the ACL will be stored.	AUTONOMYMETADATA
ACLReadersFieldCSVs	Takes a configurable field (or fields) for the Note level access control list. If you have multiple fields for access control, you can list them using comma separated values in the usual order of precedence. e.g.: ACLReadersFieldCSVs=\$Authors,\$Readers,Readers where \$Authors have greater priorities than \$Readers who in turn have greater priorities than Readers.	\$Readers

LDAP type usernames

When using LDAP compliant usernames, you must make sure that it is of the form that Notes understands. For example, Notes can have users of the forms:

“Jo Bloggs”

“CN=Jo Bloggs/O=mycompany”

“CN=Jo Bloggs/OU=marketing/O=mycompany”

The exact string needs to be formed for the security comparison to take place successfully.

The security DLL checks for both cases <USERNAME> and CN=<USERNAME> to allow for both cases. Where additional Organisation Unit (OU) and/or Organisation (O) name are supplied, you must make sure that these get sent through from whichever front end module you are using.

So for example in Portal In A Box, you can set the username for the Notes Database security to be “Jo Bloggs/OU=marketing/O=mycompany”.

ODBC/ORACLE Security - Unmapped

ODBC

Implementation

The ODBC/ORACLE Security DLL has been written so that it access a stored function or procedure to do the entitlement checking. You must write this function/procedure yourself and configure the DRE to let it know how the function is called.

Set up

To implement ODBC/ORACLE security in the DRE please follow these steps:

- 1 Make sure you have DRE version 3.1.1 or higher
- 2 Make sure you have an Oracle Client/ODBC Driver installed on the same machine as the DRE.
- 3 Copy the ODBC/ORACLE security DLL supplied by Autonomy Systems Limited in the DRE directory. This DLL can have any name, e.g. odbc_oracle_security.dll
- 4 Add a security entry for the DRE database(s) in DRE.INI. For example:

```
[Databases]
NUMDBS=1
O=ARCHIVE
```

[ARCHIVE]

Security=ODBC_ORACLE

You need to add a Security= entry under the database name. The value of this entry is the name of the security section, in this example [ODBC_ORACLE].

[ODBC_ORACLE]

```
Library=oracle_security.dll
(or Library=odbc_security.dll)
Type=AUTONOMY_SECURITY_ODBC
(or type=AUTONOMY_SECURITY_ORACLE)
Username=sys
Password=pass
Domain=autonomyoracle2
SQLStatement=select actest(%USERNAME%,%REFERENCE%)
from dual
```

ENTRY	DESCRIPTION	DEFAULT
Username	Username of the account to use for logging in to the SQL database	Empty String
Password	Password to correspond to the Username	Empty String
Domain	For ODBC: ODBC Data Source Name For Oracle: the Oracle Net Service Name	Empty String
SQLStatement	The SQL statement that performs the ACL check The SQL stored function/procedure called by the SQL Statement is used to check the access rights of the user. You must write this stored function yourself depending on the nature of your data and database. This will vary from system to system.	Empty String

The example shows a select statement that calls a function called "acltest" with two parameters %USERNAME% and %REFERENCE%. These parameters get replaced with the user's username and the document reference respectively.

Other entities you can use are %PASSWORD%, %DOMAIN%, %GROUP%. You can use these if they are required by the ACL check function/procedure. An example of a SQLStatement that uses a stored procedure instead of a function is "exec acltest '%USERNAME%', '%REFERENCE%'"

Here is a sample SQL stored function:

```
CREATE OR REPLACE FUNCTION acltest (username IN VARCHAR2, docid IN VARCHAR2)
RETURN NUMBER
IS
    usernameReturn NUMBER(5);
    CURSOR username_cursor IS
    select * from test;
    username_val username_cursor%ROWTYPE;
BEGIN
    usernameReturn := 0;
    FOR username_val in username_cursor
    LOOP
        IF username_val.username = username AND username_val.docid = docid IN
        THEN
            usernameReturn := 1;
        END IF;
    END LOOP;
```

```
RETURN (usernameReturn);  
END;  
/
```

This function looks at a table called "test" which contains (username, reference) entries, so that when you call the function:

```
select acltest("bob", "referenceBob") from dual;
```

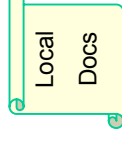
if the entry "bob", "referenceBob" is a row in the table it would return true.

When you write the stored function/procedure you must make sure that it returns a table with one column and one row containing either a number greater than 1 (ACCESS GRANTED) or 0 (ACCESS DENIED).

Oracle/ODBC Document References

The Oracle/ODBC security DLL has been designed in a flexible way so that references to the document can be of any form that can be processed by the stored procedure.

Microsoft NT Security – Unmapped



Implementation

The Notes Security DLL follows these steps to ascertain whether a user is allowed to see a document:

1. It uses the username and password to log on as that account and impersonate the user.
2. Once it is successful it attempts to open up the file on the file system.
3. If it is successful, access is granted.

Setup

This DLL has been tested on NTFS and FAT32 file systems. The DRE and the files that the DRE is going to check security access against must be on an NT server. Checking security access on Windows9X machines has proved to be unreliable at the Microsoft API level.

To implement NT security in the DRE please follow these steps:

1. Make sure you have DRE version 3.1.1 or higher.
2. On the NT server where the DRE is going to be running: log in to the NT server **locally** (not in the domain). Go to "User Manager for Domains" and select from the menu "Policies/User Rights". Tick the "Show Advanced User Rights" box and select from the drop down box the entry "Act as part of the Operating System". Add the user that the DRE is going to be running as into this list. Then select from the drop down box the entry "Log on as a batch job". Add the users which are going to be authenticated into this list (you can also add group(s) that contain the users which will be authenticated). This is required because the authentication in the dll uses a "Logon User" function in the windows API.
3. Make sure all files to be authenticated are accessible locally or over the network either via mapped drives or full UNC paths.
4. Copy the NT security DLL supplied by Autonomy Systems in the DRE directory. This DLL can have any name, e.g. NT_security.dll
5. Add a security entry for the DRE database(s) in DRE.INI. For example:

```
[Databases]
NUMDBS=1
O=ARCHIVE
```

```
[ARCHIVE]
Security=NT
```


[NT]
Library=nt_security.dll
Type=AUTONOMY_SECURITY_NT
DirLevelSecurity=1

ENTRY	DESCRIPTION	DEFAULT
DirLevelSecurity	If set to 1 it indicates that the security restrictions are set at the directory level, rather than at the document level. This is useful when doing caching of security results. If the permissions are set at the directory level, then the cache only needs to store information about the user and the directory that that user is allowed to see, instead of storing a cache for that user and every individual document. The cache will be smaller and the security checking a lot faster. If the security settings in your environment follow directory level security then we strongly recommend that you turn this setting on.	FALSE = 0 = NO

NT Document References

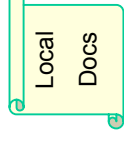
The NT security DLL has been designed to work with references which are full network paths or UNC paths to a particular file. You would normally index these files using the Autoindexer.

Troubleshooting

If you set Logging=1 you might encounter the following error message(s) in the security.log file:

NT ERROR MESSAGE		REASON
Logon failure: the user has not been granted the requested logon type at this computer.		You have not set the "Log on as a batch job" privilege for the user you are trying to authenticate.
Logon failure: A required privilege is not held by the client		You have not set the "Act as part of the Operating System" privilege for the user that the DRE is running as.
Logon failure: unknown user name or bad password		The username does not exist in the NT domain or the password given for the username is incorrect.

Microsoft NT Security - Mapped

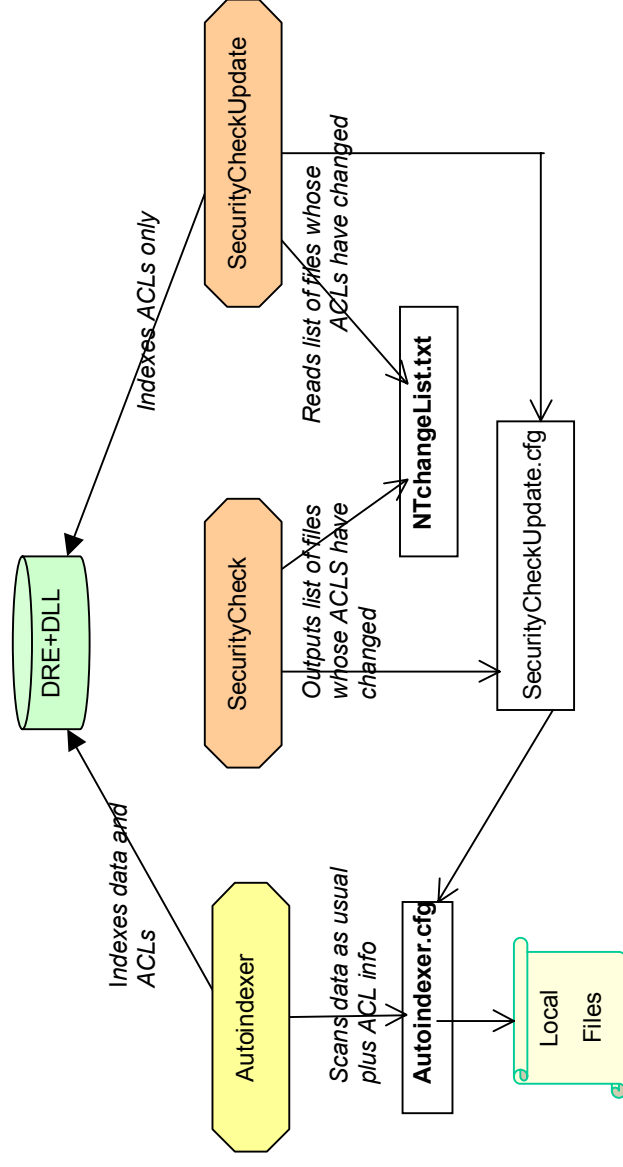


Implementation

This security setup assumes the use of the Autoindexer to index the data and a typical architecture would comprise of (please note that actual executable names might be different in a real system):

- The *DRE* which contains indexed documents in which each document will have an encrypted structured field containing the ACL information for that document (this requires DRE version 3.1.1 or above).
- The *DRE NT Security DLL* which performs the ACL comparison operation.
- The *Autoindexer* which indexes local documents into the DRE including the ACL information of those documents (this requires Autoindexer version 2.1.1 or above).
- The *SecurityCheck* process which polls the same directories that Autoindexer polls and gets notified when the ACL of any file in those directories changes. SecurityCheck then outputs its name into a text file.
- The *SecurityCheckUpdate* process which polls the output file produced by SecurityCheck and it obtains the ACL information for those files and indexes them into the DRE.

The architecture diagram is as follows:



In the diagram above, a typical control flow would be:

1. First the Autoindexer reads its config file `autoindexer.cfg`, follows its configuration and indexes the content. It also obtains the ACL information for each file and it indexes it into the DRE.
2. SecurityCheck reads its config file `SecurityCheckUpdate.cfg` (it always looks for `<executable name>Update.cfg`) which in turn points to `autoindexer.cfg`. SecurityCheck then watches all directories configured in `autoindexer.cfg` and outputs the name of any files whose ACL has been changed into `NTchangeList.txt`
3. SecurityCheckUpdate reads its config file `SecurityCheckUpdate.cfg` (the same config file is shared with SecurityCheck). It reads the output file produced by SecurityCheck, `NTchangeList.txt`, and for each file it retrieves the ACL information and indexes it into the DRE.

Installation

First of all make sure you already have the following in place:

- DRE version 3.1.1 or above
- Autoindexer version 2.1.1 or above

The installer will create a subdirectory called `MappedNTSecurity` under your existing Autoindexer directory. In this subdirectory it will install the two executables corresponding to SecurityCheck and SecurityCheckUpdate together with its configuration file. The name of the executables will be prefixed with the name of the Autoindexer executable. **These two processes will be installed as services, please make sure that you manually go to the Services panel and assign them an administration account for them to run with. They must have administrative access to work successfully.**

The installation script will install and configure the `SecurityCheckUpdate.cfg` file. It will also edit the Autoindexer configuration file to include the security settings under the [Default] section. So, if the NT security applies to all jobs in the Autoindexer then no change is needed. However, if some jobs do not require NT security then you will need to correct the configuration file yourself. By default, on startup it will spawn the two processes described above SecurityCheck and SecurityCheckUpdate if they are not already running.

The installation script will not edit the DRE configuration file, so you will need to update it manually according to the instructions outlined below.

Component Features and Configuration

It is important to remember that the Autoindexer, SecurityCheck and SecurityCheckUpdate processes must run with administrative access and must be also given access to the polled directories. The DRE does not need access to neither the ACLs nor the directories/files themselves.

Dynamic Reasoning Engine

Features:

- The DRE holds the ACL information in a structured field which is both hidden and encrypted. Hidden means that it is only used internally, it is not indexed in any way and it is never shown in result sets.
- The DRE receives as parameters, the NT domain name, the username corresponding to the user trying to access the data AND the group which the user belongs to. Password information for the user is NOT required. The DRE then calls the security DLL which forms strings of the form: <DOMAIN NAME>/<USER NAME> and <DOMAIN NAME>/<GROUP NAME> to be consistent with information store in NT ACLs. These strings are then compared to the security structured field held in the DRE.

Configuration:

To configure the DRE you specify the security details per database by setting *Type= AUTONOMY_SECURITY_NT_MAPPED* and a *ReferenceField* which is the field where the ACL information is to be stored. This can be changed to anything you want, but you'll need to make sure that the name is reflected in the other components as well. By default it is called *AUTONOMYMETADATA*. *HideField=1* is set for security purposes, so that the field containing the ACL information is never returned as part of a result in a search.

For example:

```
[Databases]
NUMDBS=1
O=MyDB

[MyDB]
Security=NTSec
```

[NTSec]
Library=nt_mapped_security.dll
Type= AUTONOMY_SECURITY_NT_MAPPED
ReferenceField=AUTONOMYMETADATA
HideField=1

[Fields]
NumFields=3
0=DRREFERENCE
1=INDEX10.DRETITLE
2=AUTONOMYMETADATA

In the [Fields] section you must have a field listed which is the same as the value in *ReferenceField*.

Parameters:

The parameters required to be sent to the DRE are:

user=<username>&domain=<domain>&group=<group>

Autoindexer

Features:

- The Autoindexer indexes the initial ACL information at the same time the content is indexed into the DRE. This is because SecurityCheck can only detect changes in the ACLs since the time it was started up.
- If SecurityCheck has not been running for a while, and some ACLs have been changed in some files, rather than indexing all of the data again you can configure Autoindexer to scan the directories and reindex only the ACLs of all the files (PollingAction=10).
- On startup the Autoindexer will automatically run the services for SecurityCheck and SecurityCheckUpdate if they are not running already. This is to ensure that the whole system is in sync and runs successfully.
- This type of security has been implemented for the “Directory Polling” method only due to the NT capability to watch directories for NT ACL changes. It might be possible for it to be implemented when performing “File Polling”, but more configuration will be required and its feasibility depends on the system setup, please contact a Technical Consultant to discuss details.
- The Autoindexer must run with Read Access Rights for the directories involved.

Configuration:

Per Job you can configure:

MappedNTSecurity=1
DREFieldName=AUTONOMYMETADATA
NTSecurityPath=.\\MappedNTSecurity\\

In order to have the SecurityCheck and SecurityCheckUpdate processes spawned on startup you can set:

SpawnSecurityProcesses=1 in [Default]

CONFIG ENTRY	DESCRIPTION	DEFAULT
MappedNTSecurity	Turns the security on	FALSE = 0 = NO
DREFieldName	Indicates the structured field in the DRE where the ACL will be stored	AUTONOMYMETADATA
NTSecurityPath	Points to the directory where SecurityCheck and SecurityCheckUpdate reside.	.\\MappedNTSecurity\\

SecurityCheck and SecurityCheckUpdate

Features:

- SecurityCheck and SecurityCheckUpdate both share the same configuration file.
- SecurityCheck monitors changes in the ACL settings of the files in the polled directories. As soon as a change is registered, the name of the file is written out to an output file. The output file will also contain the name of the Job that monitored that directory, so that SecurityCheckUpdate can find the right settings for indexing the ACL.
- SecurityCheckUpdate reads new entries from the output file produced by SecurityCheck and for each entry, it retrieves the ACL of that file and indexes it into the DRE.

Configuration:

The configuration is very similar to the Autoindexer configuration file. Below is a sample configuration file. The entries in **bold** are the ones used by both the SecurityCheck and SecurityCheckUpdate, the entry in *italics* is the one used by the SecurityCheck process only. The rest is used by the SecurityCheckUpdate only.

```
[Configuration]
// 1: file
// 2: directory
PollingMethod=1
// In milliseconds (0 to do only once)
PollingPeriod=1000
RemoveLogFileOnStart=ON
MaxLogKBytes=10000

Number=1
0=Job0

[NTSecurity]
RemoveOutputQueueFileOnStart=ON
ScanConfigFile=..\scan.cfg
```

Appendix B: DRE Security Modules

[Default]
PollingAction=10
PollingMaxNumber=100
FilePollFilename=ntchangelist.txt

[Job0]

CONFIG ENTRY	DESCRIPTION	DEFAULT
RemoveOutputQueueFileOnStart	If turned on, SecurityCheck removes the output file on startup.	FALSE
ScanConfigFile	The Autoindexer configuration file	Empty String
PollingAction=10	Index ACL action	NONE
FilePollFilename	The output filename. SecurityCheck writes to it, while SecurityCheckUpdate reads from it.	NONE Empty String

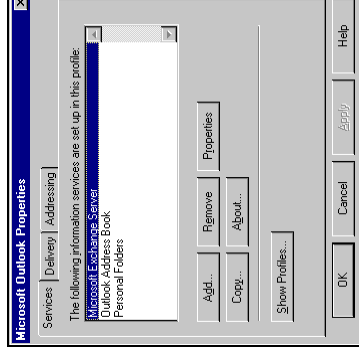
Microsoft Exchange Security – Unmapped



This DLL has been tested on Exchange Server 5.5, with MAPI Clients Outlook97 and Outlook2000. The server where the DRE is running from must have a MAPI Client installed and it must be able to communicate with the Exchange server. The easiest way of achieving this is to install a copy of Outlook (97, 98 or 2000) and make sure that you install the Exchange client connection option.

Preliminary Check

To check if the PC is able to connect to the Exchange server look in the control panel for an icon labelled "Mail and Fax" (on Win2000 systems this is labelled just "Mail" . On Win95, 98 or NT if this is called just "Mail" then you do not have the Exchange client software installed). Click on this and check that you can see a tab named "services" and that it contains a service called "Microsoft Exchange Server".



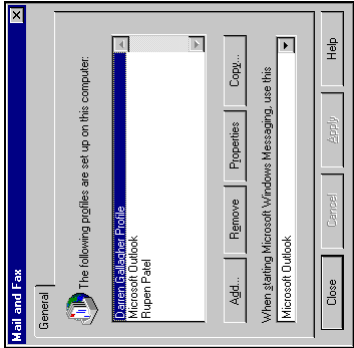
If you cannot see this tab then you need to reinstall outlook and ensure that you install the Exchange client connection option.

For the security dll to work correctly the program must be run using the NT Domain Administrator Account. (The DRE service can be configured to run as a particular user in the service control panel).

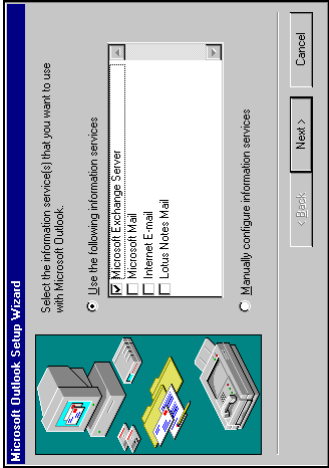
To check that you have the required permissions create a test profile that points to a mailbox on the Exchange server. Click on the "Show Profiles.." button.

Appendix B: DRE Security Modules

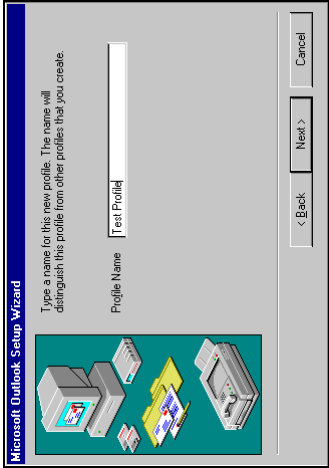
Now click on the "Add" button.



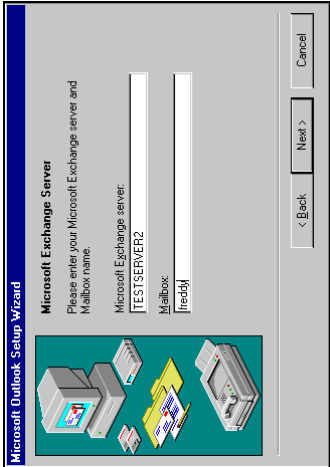
Select the Microsoft Exchange Server information service from the list and click "Next".



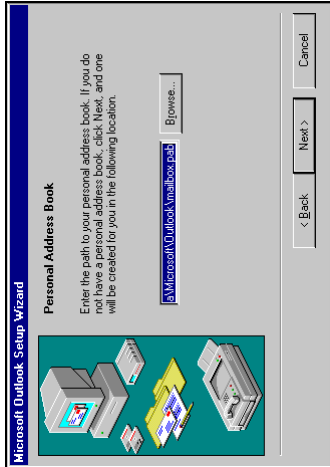
Enter the name of your profile and click "Next".



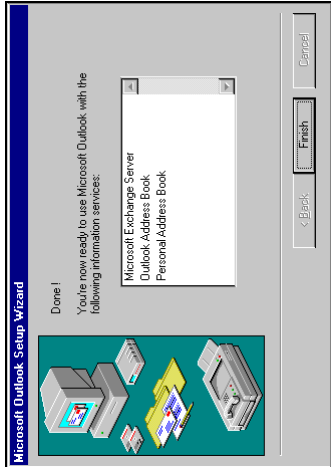
Enter the name of your Exchange server and the mailbox that you wish to connect to. Click "Next". The next screen will ask if you travel with the computer. Select "no" and click on "Next".



Select the location of the the personal address book and click "Next" .

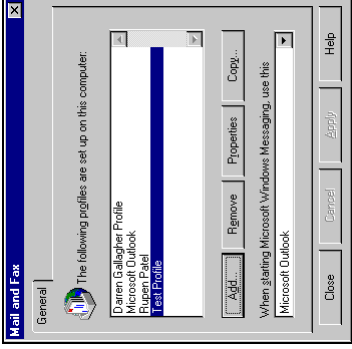


If the profile was created successfully then you will see this screen. Click "Finish" and check that the new profile is listed in the list of profiles.



Appendix B: DRE Security Modules

We now know that we have permissions to create Profiles that point to mailboxes on the Exchange server. The profile that we created can now be deleted by selecting it and clicking on the "Remove" button. If the profile was successfully created then continue to the next section.



Setup

To implement Exchange security in the DRE please follow these steps:

- 1 Make sure you have DRE version 3.1.1 or higher.
- 2 Copy the Exchange security DLL supplied by Autonomy Systems in the DRE directory. This DLL can have any name, let's say it's call Exchange_security.dll
- 3 Add a security entry for the DRE database(s) in DRE.INI. For example:

```
[Databases]
NUMDBS=1
0=ARCHIVE
```

```
[ARCHIVE]
Security=Exchange
```

You need to add a Security= entry under the database name. The value of this entry is the name of the security section, in this example [NT].

[Exchange]
Library=exchange_security.dll
Type=AUTONOMY_SECURITY_EXCHANGE
Profile=Jo_Bloggs

- 4 ExchangeFetch must have the setting AppendFolderPath=1 for the security to work.

ENTRY	DESCRIPTION	DEFAULT
Profile	Profile username of the account to use for accessing ACL information in the Exchange Server. This profile must have administrative access.	Empty String

Exchange Document References

The Exchange security DLL has been designed to work with references of documents obtained with ExchangeFetch.

The values of DRREFERENCE of the documents in the DRE have to be of the following form:

<http://myserver/exchange/forms/ipm/note/read.asp?command=open&obj=000000001A447390AA6611CD9BC800AA002FC45A09002418AD014DA3D311B2E40090279B6AB500000000278C00002418AD014DA3D311B2E40090279B6AB5000000004F880000&timedout=1&FolderPath=IPM%5FSUBTREE%5C>
Sarah

Exchange Security - Mapped



Implementation

This security setup assumes the use of the ExchangeFetch to extract and index Exchange content into a DRE. The implementation components would be as follows:

- The *DRE* contains indexed documents in which each document will have an encrypted structured field containing the ACL information for that document (this requires DRE version 3.1.1 or above).
- The *DRE Exchange Security DLL* which performs the ACL comparison operation.
- The *ExchangeFetch* which indexes Exchange Server documents into the DRE including the ACL information of those documents (this requires ExchangeFetch version 2.1.1 or above).

Component Features and Configuration

First of all make sure you already have the following in place:

- DRE version 3.1.1 or above
- ExchangeFetch version 2.1.1 or above

Dynamic Reasoning Engine

Features:

- The DRE holds the ACL information in a structured field which is both hidden and encrypted. Hidden means that it is only used internally, it is not indexed in any way and it is never shown in result sets.
- The DRE receives as parameters, the Exchange Profile user name AND the group which the user belongs to, together with the Domain. Password information for the user is NOT required.

Configuration:

To configure the DRE you specify the security details per database by setting *Type= AUTONOMY_SECURITY_EXCHANGE_MAPPED* and a *ReferenceField* which is the field where the ACL information is to be stored. This can be changed to anything you want, but you'll need to make sure that the name is reflected in the other components as well. By default it is called *AUTONOMYMETADATA*. *HideField=1* is set for security purposes, so that the field containing the ACL information is never returned as part of a result in a search.

For example:

```
[Databases]
NUMDBS=1
0=MyDB

[MyDB]
Security=ExchangeSec

[ExchangeSec]
Library=exchange_mapped_security.dll
Type= AUTONOMY_SECURITY_EXCHANGE_MAPPED
ReferenceField=AUTONOMYMETADATA
HideField=1

[Fields]
NumFields=3
0=DREREFERENCE
1=INDEX10:DRETITLE
2=AUTONOMYMETADATA
```

In the [Fields] section you must have a field listed which is the same as the value in *ReferenceField*.

Parameters:

The parameters required to be sent to the DRE are:

user=<username>&group=<group>&domain=<domain>

ExchangeFetch

Features:

- The ExchangeFetch indexes the initial ACL information at the same time the Exchange Server content is indexed into the DRE. At any subsequent cycle the ExchangeFetch will detect changes in the ACL, as well as in the content, and update them in the DRE.
- The ExchangeFetch must run with a profile of a user who has administrative privileges, this is because it needs to have access to the ACL information stored within Exchange.

Configuration:

In order to implement mapped security within Exchange, the ExchangeFetch must have 2 jobs configured which work together. The first job indexes the data from the Exchange Server as usual together with the ACLs for each document. The second job is in charge of only detecting any changes in the ACLs of the documents on the Exchange Server. These ACLs are then extracted and indexed into the DRE.

An example configuration is as follows:

[Fetch]

FetchName0=DATAINDEX

FetchName1=ACLUPDATE

[DATAINDEX]

ProfileName=Admin

StartFolder=IPM_SUBTREE\data

ReferencePrefix=http://testserver/exchange/forms/ipm/note/read.asp?command=open&obj=

ReferencePostfix=&timedout=1

FetchRepeatInterval=900

StatFile=DATAINDEX.stat

ACLStatFile=DATAINDEXACL.stat

ACLFieldName=AUTONOMYMETADATA

IndexACLs=true

[ACLUPDATE]

ProfileName=Admin

StartFolder=IPM_SUBTREE\data

ReferencePrefix=http://testserver/exchange/forms/ipm/note/read.asp?command=open&obj=

ReferencePostfix=&timedout=1

FetchRepeatInterval=60

ACLStatFile=DATAINDEXACL.stat

ACLFieldName=AUTONOMYMETADATA

UpdateACLs=true

Note the different RepeatIntervals for each fetch job. In this case we get the desired affect of content being updated as frequently as is necessary BUT the security ACL info is updated much more often. These values should reflect the amount of content that exists on the Exchange Server, how long it takes to index initially, how often it changes and how often the security is updated on average.

Also, note that the value for ACLStatFile must point to the same file for both jobs.

ENTRY	DESCRIPTION	DEFAULT
ACLFieldName	The DRE field where the folder ACL is indexed	AUTONOMYMETADATA
StatFile ACLStatFile	StatFile stores the items it has found while ACLStatFile stores the folder ACLS in order to check for updates in the future. It uses ACLStatFile to determine which ACLs have been updated since the last run in order to reindex them into the DRE.	<Job Name>.stat ACL.stat
IndexACLs	Instructs the ExchangeFetch to index document ACLs into a DRE field. This is used only in the 1st fetch job.	FALSE = 0 = NO
UpdateACLs	Instructs the ExchangeFetch to detect the documents whose ACLs have been updated and it reindexes ONLY those ACLs into the DRE.	FALSE = 0 = NO
DebugACLs	Used to debug the fetch while it is fetching folder ACLs. It creates 2 new log files to help you see what it is doing.	FALSE

Appendix B: DRE Security Modules

Email Template Configuration:

You need to add a line to the email.tmpl file to reflect the ACL Field to be indexed:

```
#DREFERENCE <!--ATNMYTAGMAPIDreRef-->
#DRETITLE <!--ATNMYTAGMAPISubject-->
#DRECONTENT
<!--ATNMYTAGMAPIContent-->
#DREDATE <!--ATNMYTAGMAPILastModifiedDate-->
#DREFIELD AUTONOMYMETADATA="<!--ATNMYTAGMAPIFOLDERACL-->"
#DREDDDOC
```

The name **AUTONOMYMETADATA** field name can be changed to whatever name is specified in the DRE and ACLFieldName, as long as they are all the same.

Appendix C: Service Settings for use with Autonomy DiSH

x behaves as a standard Autonomy Service and can be used in conjunction with DiSH.

If the following settings are added to the configuration file, the service port is enabled and will accept the standard status and control commands.

[Service]	Description	Default
SERVICEPORT	DiSH listens for commands on this port	None
SERVICECONTROLCLIENTS	This specifies those who have control by wildcard IP match	None
SERVICESTATUSCLIENTS	This specifies those who can get the status of services by wildcard IP match	None

Standard Status and Control commands on the service port:

ACTION	Control/ Status	DESCRIPTION
GETSTATUS	STATUS	This is used to find out if a service is running
GETSTATUSINFO	STATUS	This is used to find out the product name, version etc.
GETCONFIG	STATUS	This is used to obtain the configuration file
SETCONFIG	CONTROL	This is used to set the configuration file
GETSTATISTICS	STATUS	This is used to gather statistics from the service
START	CONTROL	This is used to start a service
STOP	CONTROL	This is used to stop a service
PAUSE	CONTROL	This is used to pause a service
MERGECONFIG	CONTROL	Merge the fragment of the config file passed with the services config file.