**EMC/Documentum xPlore 1.3**

**Overview, Installation,**

**Configuration and Maintenance (SOP)**

Contents

[1 Overview 5](#_Toc356106344)

[1.1 Document Purpose 5](#_Toc356106345)

[1.2 Revision History 5](#_Toc356106346)

[2 Introduction to xPlore 6](#_Toc356106347)

[2.1 xPlore Glossary 6](#_Toc356106348)

[2.2 Features 7](#_Toc356106349)

[2.2.1 Indexing features 8](#_Toc356106350)

[2.2.2 Search features 8](#_Toc356106351)

[2.2.3 Administration features 8](#_Toc356106352)

[2.2.4 Limitations 9](#_Toc356106353)

[2.3 Architectural overview 13](#_Toc356106354)

[2.3.1 Physical architecture 13](#_Toc356106355)

[2.3.2 xPlore disk areas 14](#_Toc356106356)

[2.3.3 xPlore instances 15](#_Toc356106357)

[2.3.4 xDB libraries and Lucene index 16](#_Toc356106358)

[2.4 Logical architecture 19](#_Toc356106359)

[2.5 Managing the System 27](#_Toc356106360)

[3 Installation anD configuration 31](#_Toc356106361)

[3.1 Configuring xPlore instance 31](#_Toc356106362)

[Finalize Primary Instance configuration 32](#_Toc356106363)

[Restart xPlore instance 34](#_Toc356106364)

[3.2 Configure Index Agents. 39](#_Toc356106365)

[3.2.1 Configure xPl-A IA 40](#_Toc356106366)

[4. Restart xPlore instance and Index Agents on xPl-A 43](#_Toc356106367)

[At this point, Primary DSearch and all configured IndexAgents are running, but Index Agents are stopped. Visit URL for DSearch Admin console and each IA console and verify. Do not start Index agents in UI. 43](#_Toc356106368)

[3.2.2 Configure IA on xPl-B 44](#_Toc356106369)

[Note: installing IA on xPL-B host is similar, but not identical to installing IA on xPl-B. 44](#_Toc356106370)

[1. On CS-A run: 44](#_Toc356106371)

[6. Update FT related object in docbase. On CS-A run: 47](#_Toc356106372)

[cd /opt/dmadmin/CS67SP1/product/6.7SP1/install/admin 47](#_Toc356106373)

[3.3 Back-up xPlore. 48](#_Toc356106374)

[4 Initial indexing 51](#_Toc356106375)

[4.1 xPl-A 51](#_Toc356106376)

[4.2 xPl-B 58](#_Toc356106377)

[5 Maintainance 60](#_Toc356106378)

[5.1 Enabling indexing for an object type 60](#_Toc356106379)

[5.2 Registering a type for full-text indexing 60](#_Toc356106380)

[5.3 Reindexing 60](#_Toc356106381)

[5.4 Monitoring xPlore 62](#_Toc356106382)

[5.4.1 Watchdog service 62](#_Toc356106383)

[5.4.2 DA 63](#_Toc356106384)

[Index Agent. 63](#_Toc356106385)

[Go through all docbases which are configured for Full Text Indexing: 63](#_Toc356106386)

[5.5 Verifying integrity of indexes. 65](#_Toc356106387)

[5.6 Re-directing FT Searches to a standby xPlore host. 67](#_Toc356106388)

[5.7 Missing Indexes. Identifying docbase content which has no corresponding indexes in xPlore. 68](#_Toc356106389)

[5.8 File System Backup and Restore 71](#_Toc356106390)

[5.8.1 Backup 71](#_Toc356106391)

[To perform FS xPlore backup, follow these steps: 71](#_Toc356106392)

[5.8.2 Restore. 73](#_Toc356106393)

[5.9 Backing up and restoring xPlore Federation. 74](#_Toc356106394)

[5.9.1 Backup. 74](#_Toc356106395)

[5.9.2 Restore. 74](#_Toc356106396)

[5.10 Restoring data after an instance fails 75](#_Toc356106397)

[5.11 Testing search 77](#_Toc356106398)

[5.12 Removing entries from the index 77](#_Toc356106399)

[5.13 Document processing (CPS) reports 78](#_Toc356106400)

[6 COB SETUP 79](#_Toc356106401)

[6.1 Failing-over from Prod to COB 79](#_Toc356106402)

[6.2 Failing over from COB to Prod. 83](#_Toc356106403)

[7 APPEnDIX A. xplore installation example 84](#_Toc356106404)

[8 APPEnDIX B. xplore Start/stop scripts 88](#_Toc356106405)

[8.1 Dsearch Admin console. Primary instance 88](#_Toc356106406)

[8.2 Index Agents 88](#_Toc356106407)

[8.3 Watchdog 88](#_Toc356106408)

[9 APPEnDIX c. xplore logs 89](#_Toc356106409)

[10 APPEnDIX D. External CPS. 91](#_Toc356106410)

# Overview

## Document Purpose

This purpose of this document is to provide detailed procedure on indexing 67SP1 Docbases in xPlore 1.3 HA deployment.

## Revision History

| **Date** | **Version** | **Author** | **Change**  **(Topic, Section, Page)** |
| --- | --- | --- | --- |
| 2.24.2013 | 1.0 | Lev Jacobson  [lev.jacobson@citi.com](mailto:lev.jacobson@citi.com)  1 908 563 3296 | Initial release |

# Introduction to xPlore

## xPlore Glossary

|  |  |
| --- | --- |
| **Term** | **Description** |
| category | A category defines a class of documents and their XML structure. |
| collection | A collection is a logical group of XML documents that is physically stored in an xDB library. A collection represents the most granular data management unit within xPlore. |
| content processing service (CPS) | The content processing service (CPS) retrieves indexable content from content sources and determines the document format and primary language. CPS parses the content into index tokens that xPlore can process into full-text indexes. |
| domain | A domain is a separate, independent group of collections with an xPlore deployment. |
| DQL | Documentum Query Language, used by many Content Server clients |
| FTDQL | Full-text Documentum Query Language |
| ftintegrity | A standalone Java program that checks index integrity against Content Server repository documents. The ftintegrity script calls the state of index job in the Content Server. |
| full-text index | Index structure that tracks terms and their occurrence in a document. |
| index agent | Documentum application that receives indexing requests from the Content Server. The agent prepares and submits an XML representation of the document to xPlore for indexing. |
| ingestion | Process in which xPlore receives an XML representation of a document and processes it into an index. |
| instance | A xPlore instance is one deployment of the xPlore WAR file to an application server container. You can have multiple instances on the same host (vertical scaling), although it is more common to have one xPlore instance per host (horizontal scaling). The following processes can run in an xPlore instance: CPS, indexing, search, xPlore administrator. xPlore can have multiple instances installed on the same host. |
| lemmatization | Lemmatization is a normalization process in which the lemmatizer finds a canonical or dictionary form for a word, called a lemma. Content that is indexed is also lemmatized unless lemmatization is turned off. Terms in search queries are also lemmatized unless lemmatization is turned off. |
| Lucene | Apache open-source, Java-based full-text indexing, and search engine. |
| node | In xPlore and xDB, node is sometimes used to denote instance. It does not denote host. |
| persistence library | Saves CPS, indexing, and search metrics. Configurable in indexserverconfig.xml. |
| state of index job | Content Server configuration installs the state of index job dm\_FTStateOfIndex. This job is run from Documentum Administrator. The ftintegrity script calls this job, which reports on index completeness, status, and indexing failures. |
| status library | A status library reports on indexing status for a domain. There is one status library for each domain. |
| stop words | Stop words are common words filtered out of queries to improve query performance. Stop words can be searched when used in a phrase. |
| text extraction | Identification of terms in a content file. |
| token | Piece of an input string defined by semantic processing rules. |
| tracking library | An xDB library that records the object IDs and location of content that has been indexed. There is one tracking database for each domain. |
| transactional support | Small in-memory indexes are created in rapid transactional updates, then merged into larger indexes. When an index is written to disk, it is considered clean. Committed and uncommitted data before the merge is searchable along with the on-disk index. |
| watchdog service | Installed by the xPlore installer, the watchdog service pings all xPlore instances and sends an email notification when an instance does not respond. |
| xDB | xDB is a database that enables high-speed storage and manipulation of many XML documents. In xPlore, an xDB library stores a collection as a Lucene index and manages the indexes on the collection. The XML content of indexed documents can optionally be stored. |
| XQFT | W3C full-text XQuery and XPath extensions described in [XQuery and XPath Full Text 1.0](http://www.w3.org/TR/xpath-full-text-10/). Support for XQFT includes logical full-text operators, wildcard option, anyall option, positional filters, and score variables. |
| XQuery | W3C standard query language that is designed to query XML data. xPlore receives xQuery expressions that are compliant with the [XQuery](http://www.w3.org/TR/xquery/) standard and returns results. |

## Features

Documentum xPlore is a multi-instance, scalable, high-performance, full-text index server that can be configured for high availability and disaster recovery.

The xPlore architecture is designed with the following principles:

* Uses standards as much as possible, like XQuery
* Uses open source tools and libraries, like Lucene
* Supports enterprise readiness: High availability, backup and restore, analytics, performance tuning, reports, diagnostics and troubleshooting, administration GUI, and configuration and customization points.
* Supports virtualization, with accompanying lower total cost of ownership.

## Indexing features

* Collection topography: xPlore supports creating collections online, and collections can span multiple file systems.
* Transactional updates and purges: xPlore supports transactional updates and purges of indexes as well as transactional commit notification to the caller.
* Multithreaded insertion into indexes: xPlore ingestion through multiple threads supports vertical
* scaling on the same host.
* Dynamic allocation and deallocation of capacity: For periods of high ingestion, you can add a CPS instance and new collection. Add content to this collection, then move the collection to another instance for better search performance. You can then decommission the CPS instance.
* Temporary high query load: For high query load, like a legal investigation, add an xPlore instance for the search service and bind collections to it in read-only mode.
* Growing ingestion or query load: If your ingestion or query load increases due to growing business, you can add instances as needed.
* Extensible indexing pipeline using the open-source UIMA framework.
* Configurable stop words and special characters.

## Search features

* Case sensitivity: xPlore queries are lower-cased (rendered case-insensitive).
* Full-text queries: To query metadata, set up a specific index on the metadata.
* Faceted search: Facets in xPlore are computed over the entire result set or over a configurable number of results.
* Security evaluation: When a user performs a search, permissions are evaluated for each result. Security can be evaluated in the xPlore full-text engine before results are returned to Content Server, resulting in faster query results. This feature is turned on by default and can be configured or turned off.
* Native XQuery syntax: The xPlore full-text engine supports XQuery syntax.
* Thesaurus search to expand query terms.
* Fuzzy search finds misspelled words or letter reversals.
* Boost specific metadata in search results.
* Extensive testing and validation of search on supported languages.

## Administration features

* Multiple instance configuration and management.
* Reports on ingestion metrics and errors, search performance and errors, and user activity.
* Collections management: Creating, configuring, deleting, binding, routing, rebuilding, querying.
* Command-line interface for automating backup and restore.

## Limitations

***ACLs and aspects are not searchable by default***

ACLs and aspects are not searchable by default, to protect security. You can reverse the default by editing indexserverconfig.xml. Set full-text-search to true in the sub-path definition for acl\_name and r\_aspect\_name and then reindex your content.

***Ingestion of many large files can cause failures***

When CPS processes two or more large files at the same time, the CPS log file reports one of the following errors (cps\_daemon.log):

ERROR [Daemon-Core-(3400)] Exception happened, ACCESS\_VIOLATION,

Attempt to read data at address 1 at (connection-handler-2)

...

FATAL [DAEMON-LP\_RLP-(3440)] Not enough memory to process linguistic requests.

Error message: bad allocation

Workaround using xPlore administrator. Select an instance and click **Configuration**. Change the following to smaller values:

• Batch size: Decrease batch size to decrease the number of documents in a failed batch. (All

documents in a batch fail to be indexed if one document fails.)

• Max text threshold

• Thread pool size

***Only metadata is indexed if a format is not supported***

CPS cannot process certain formats as documents or email attachments, and PDF input fields. Only metadata of unsupported contents is indexed during ingestion.

To see format processing errors, use the xPlore administrator report *Document Processing Error Detail* and choose *File format unsupported*.

To test whether a format is supported, try uploading it in xPlore administrator. If no error code is

reported in the Document processing error report, the format has been successfully indexed.

***CPS daemon must restart after fatal ingestion error***

CPS daemon restarts when ingestion generates a fatal error. The CPS log indicates that the connection has been reset:

2009-02-10 12:19:55,425 INFO [DAEMON-CORE-(-1343566944)]

Daemon is shutdown forcefully.

2009-02-10 12:19:55,512 ERROR [MANAGER-CPSTransporter-(CPSWorkerThread-6)]

Failed to receive the response XML.

java.net.SocketException: Connection reset

***Batch failure***

Indexing requests are processed in batches. When one request in a batch fails when the index is written to xDB, the entire batch fails.

***Collection names cannot be duplicated***

The name of an adopted collection cannot be reused, because the name still exists in the domain.

An adopted collection is a collection that has been moved to a parent collection. The adopted collection becomes a sub-collection. This kind of collection is created to boost ingestion rate, and later adopted for better search performance.

***Only one language is analyzed for indexing***

Documents with multiple languages are indexed, but only one language is selected for indexing

tokens. Words in other languages are sometimes not indexed properly.

***dftxml attributes are not indexed***

xPlore does not index attribute values on XML elements the dftxml representation of the input

document. For example, you cannot find all documents for which the value of the *dmfttype* attribute of the element *acl\_name* is *dmstring*.

***Phrase searches***

The content of a phrase search is not lemmatized.

Search fails for parts of common phrases. A common phrase like *because of*, *a good many*, or

*status quo* is tokenized as a phrase and not as individual words. A search for a word in the phrase like *because* fails.

***Some lightweight sysobjects (LWSOs) are not fully indexed***

***and searchable***

In a Documentum repository, lightweight sysobjects such as emails inherit many attributes from the parent object. If the LWSOs are not materialized, a query on the inherited attributes fails. Some Documentum client applications, such as Webtop and DCO, materialize emails so they are fully searchable.

***Skipped collections are not reported in query results***

When a collection is unavailable or corrupted, it is skipped in a query. xPlore does not notify the

user or administrator that the collection was skipped. Results can be different when the collection is restored or brought back online.

***Special characters limitations***

Special characters are treated as white space. The underscore in the following document name is treated as white space. Special characters lists are limited to ANSI characters.

## Architectural overview

xPlore provides query and indexing services that can be integrated into external content sources such as the Documentum content management system. External content source clients like Webtop or CenterStage, or custom Documentum DFC clients, can send indexing requests to xPlore.

Each document source is configured as a domain in xPlore. You can set up domains using xPlore administrator. For Documentum environments, the Documentum index agent creates a domain for each repository and a default collection within that domain.

Documents are provided in an XML representation to xPlore for indexing through the indexing APIs.

In a Documentum environment, the Documentum index agent prepares an XML representation of each document. The document is assigned to a category, and each category corresponds to one or more collections as defined in xPlore. To support faceted search in Documentum repositories, you can define a special type of an index called an *implicit composite index*.

xPlore instances are web application instances that reside on application servers. When an xPlore instance receives an indexing request, it uses the document category to determine what is tokenized and saved to the index. A local or remote instance of the content processing service (CPS) fetches the content. CPS detects the primary language and format of a document. CPS then extracts indexable content from the request stream and parses it into tokens. The tokens are used for building a full-text index.

xPlore manages the full-text index. An external Apache Lucene full-text engine is embedded into the EMC XML database (xDB). xDB tracks indexing and updates requests, recording the status of requests and the location of indexed content. xDB provides transactional updates to the Lucene index. Indexes are still searchable during updates.

When an instance receives a query request, the request is processed on all included collections, then the assembled query results are returned.

xPlore provides a web-based administration console.

## Physical architecture

The xPlore index service and search service are deployed as a WAR file to a JBoss application server that is included in the xPlore installer. xPlore administrator and online help are installed as war files in the same JBoss application server. The index is stored in the storage location that was selected during configuration of xPlore (/home/dmadmin/xPlore/data).

## xPlore disk areas

xPlore creates disk areas for xDB data and redo log, the Lucene index within xDB, a temp area, and xPlore configuration and utilities. When you run the xPlore installer to configure an index agent, a disk area is created for content staging. The following table describes how these areas are used during indexing and search.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Area** | **Description** | **Use in indexing** | **Use in search** | |
| *xplore\_home*/data | Stores dftxml, metrics,  audit, ACLs and groups.  Performs query lookup  and retrieval and facet  and security information | Next free space is  consumed by disk block  for batch XML files.  Index updated through  inserts and merges | Random access retrieval  for specific elements and  summary. Inverted index  lookup and facet and  security retrieval |
| *xplore\_home*/config/log | Stores transaction  information | Updates to xDB data are  logged | Provides snapshot  information during some  retrievals |
| *xplore\_home*/dsearch/admin  /lib | Used for restore when  xPlore is down |  |  |
| temp | 1. (CPS) Intermediate  processing  2. (CPS) Exports to the  index service  3. Index: Updates to  the Lucene index  (non-transactional) | Non-committed data is  stored to the log |  |
| Index agent content  staging area | Temporarily holds  content during indexing  process | Holds content |  |

## xPlore instances

An xPlore instance is a web application instance (WAR file) that resides on an application server. You can have multiple instances on the same host (vertical scaling), although it is more common to have one xPlore instance per host (horizontal scaling). You create an instance by running the xPlore installer.

The first instance that you install is the primary instance.

**Adding or deleting an instance.**

To add an instance to the xPlore system, run the xPlore configurator script. If an xPlore instance exists on the same host, select a different port for the new instance, because the default port is already in use.

To delete an instance from the xPlore system, use the xPlore configurator script. Shut down the

instance before you delete it.

You manage instances in xPlore administrator. Click **Instances** in the left panel to see a list of

instances in the right content pane. You see following instance information:

• OS information: Host name, status, OS, and architecture.

• JVM information: JVM version, active thread count, and number of classes loaded.

* + xPlore information: xDB version, instance version, instance type, and state.

An instance can have one or more of the following features enabled:

* + Content processing service (CPS)
  + Indexing service
  + Search service
  + xPlore Administrator (includes analytics, instance, and data management services)
  + Spare: A spare instance can be manually activated to take over for a disabled or stopped instance.

You manage an instance by selecting the instance in the left panel. Collections that are bound to the instance are listed on the right. Click a collection to go to the Data Management view of the collection.

The application server instance name for each xPlore instance is recorded in indexserverconfig.xml. If you change the name of the JBoss instance, change the value of the attribute *appserver-instance-name* on the *node* element for that instance. This attribute is used for registering and unregistering instances.

Back up the xPlore federation after you change this file.

## xDB libraries and Lucene index

xDB is a Java-based XML database that enables high-speed storage and manipulation of many XML documents. xDB supports the XQuery language and XQFT query specifications. An xDB library has a hierarchical structure like an OS directory. The library is a logical container for other libraries or XML documents.

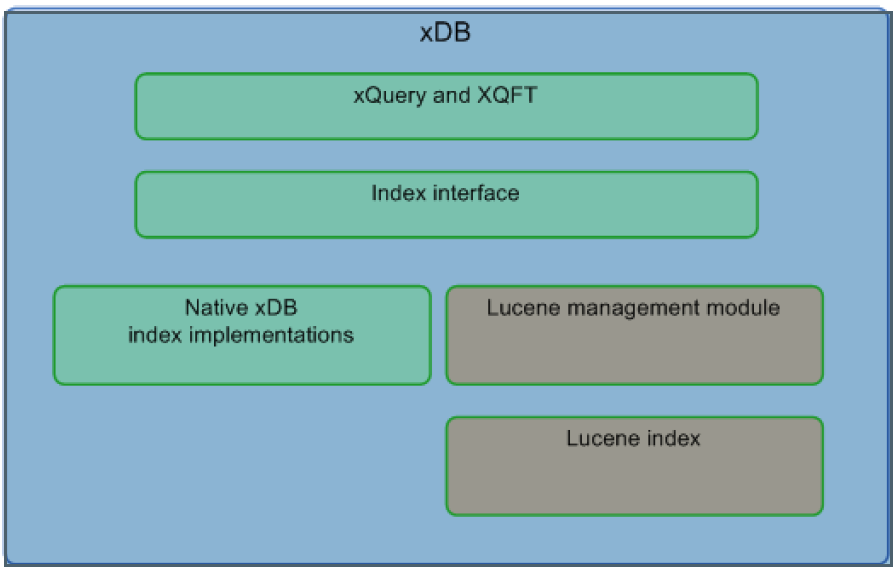
A library corresponds to a collection in xPlore with additional metadata such as category, usage, and properties. An xDB library stores an xPlore collection as one or more Lucene indexes that can include the XML content that is indexed. xPlore manages the indexes on the collection.

xDB manages the following libraries for xPlore:

* + The root library contains a SystemData with metrics and audit databases. These databases record metrics and audit queries by xPlore instance.
  + Each domain contains an xDB tracking library (database) records the content that has been indexed.
  + Each domain contains a status library (database) that reports indexing status for the domain.
  + Each domain contains one or more data libraries. The *default* library is the first that is created for a domain.

When xPlore processes an XML representation of an input document and supplies tokens to xDB, xDB stores them into a Lucene index. Optionally, xPlore can be configured to store the content along with the tokens. A tracking database in xDB manages deletes and updates to the index. When documents are updated or deleted, changes to the index are propagated. When xPlore supplies XQuery expressions to xDB, xDB passes them to the Lucene index. To query the correct index, xDB tracks the location of documents.

xDB manages parallel dispatching of queries to more than one Lucene index when parallel queries are enabled. For example, if you have set up multiple collections on different storage locations, you can query each collection in parallel.



An xDB library is stored on a data store. If you install more than one instance of xPlore, the storage locations must be accessible by all instances. The xDB data stores and indexes can reside on a separate data store, SAN or NAS. The locations are configurable in xPlore administrator. If you do not have heavy performance requirements, xDB and the indexes can reside on the same data store

.

***Indexes***

xDB has several possible index structures that are queried using XQuery. The Lucene index is

modeled as a multi-path index (a type of composite index). in xDB. The Lucene index services both value-based and full-text probes of the index.

Covering indexes are also supported. When the query needs values, they are pulled from the index and not from the data pages. Covering indexes are used for security evaluation and facet computation.

You can configure none, one, or multiple indexes on a collection. An explicit index is based on values of XML elements, paths within the XML document, path-value combination, or full-text content.

For example, following is a value indexed field:

/dmftdoc[dmftmetadata//object\_name="foo"]

Following is a tokenized, full-text field:

/dmftdoc[dmftmetadata//object\_name ftcontains ’foo’]

Indexes are defined and configured in indexserverconfig.xml.

## Logical architecture

A domain contains indexes for one or more categories of documents. A category is logically

represented as one or more collections. Each collection contains indexes on the content and metadata.

When a document is indexed, it is assigned to a category or class of documents and indexed into one of the category collections.

***Domains***

A domain is a separate, independent, logical grouping of collections with an xPlore deployment. For example, a domain could contain the indexed contents of a single Documentum content repository.

Domains are defined in xPlore administrator in the data management screen. A domain can have multiple collections in addition to the default collection.

The Documentum index agent creates a domain for the repository to which it connects. This domain receives indexing requests from the Documentum index agent.

***Categories***

A category defines how a class of documents is indexed. All documents submitted for ingestion

must be in XML format. (For example, the Documentum index agent prepares an XML version for Documentum repository indexing.) The category is defined in indexserverconfig.xml and managed by xPlore. A category definition specifies the processing and semantics that is applied to an ingested XML document. You can specify the XML elements that are used for language identification. You can specify the elements that have compression, text extraction, tokenization, and storage of tokens.

You also specify the indexes that are defined on the category and the XML elements that are not indexed. A collection belongs to one category.

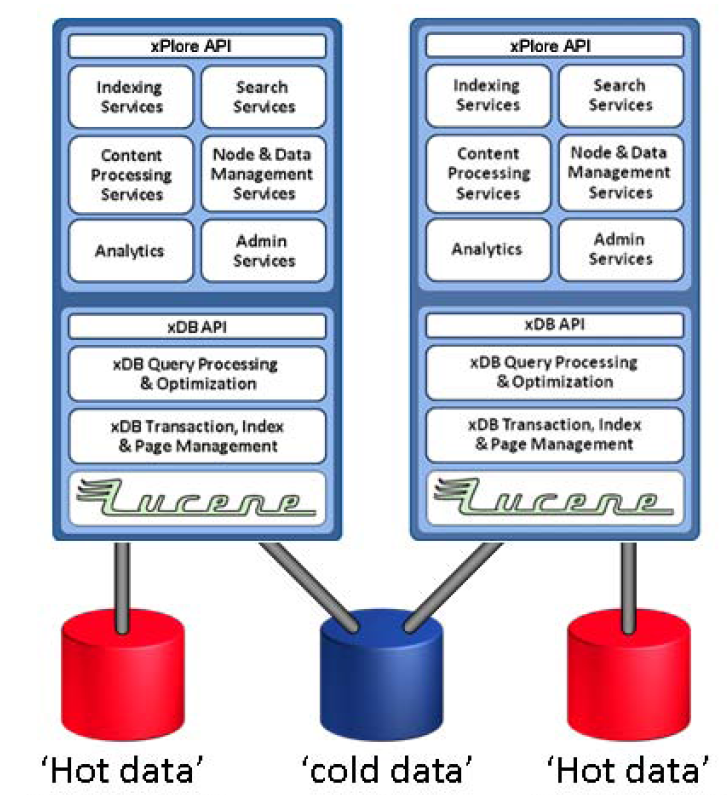
***Collections***

A collection is a logical group of XML documents that is physically stored in an xDB detachable

library. A collection represents the most granular data management unit within xPlore. All documents submitted for indexing are assigned to a collection. A collection generally contains one category of documents. In a basic deployment, all documents in a domain are assigned to a single default collection.

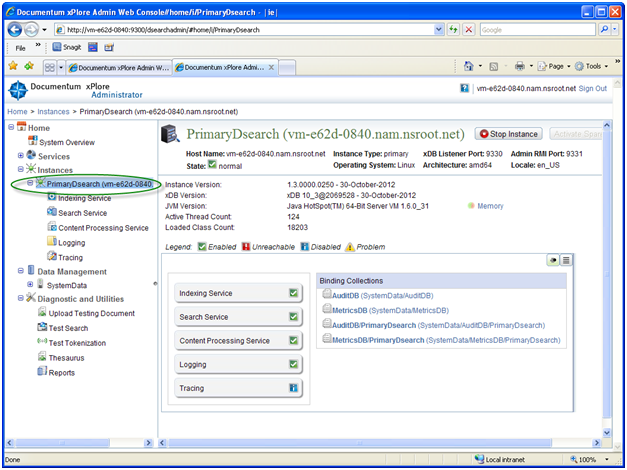
You can create sub-collections under each collection and route documents to user-defined collections.

A collection is bound to a specific instance in read-write state (index and search, index only, or update and search). A collection can be bound to multiple instances in read-only state (search-only). Three collections (two hot and one cold) with their corresponding instances are shown.



The metrics and audit systems store information in collections in a domain named SystemData. You can view this domain and collections in xPlore administrator. One metrics and one audit database is defined. Each database has a subcollection for each xPlore instance.

The following diagram shows the services of a simple xPlore system: Two installed instances, each with its own indexing, search, and CPS services.



***Documentum domains and categories***

***Repository domains***

An xPlore domain generally maps to a single Documentum repository. Within that domain, you can direct documents to one or more collections. In the following configuration in indexserverconfig.xml, a repository is mapped to a domain. Three collections are defined: one for metadata and content (default), one for ACLs, and one for groups. These latter two collections are used to filter results for permissions before returning them to the client application. The collections in the domain can be distributed across multiple xPlore instances. (Each collection is bound to an instance.)

<domain storage-location-name="default" default-document-category="dftxml"

name="TechPubsGlobal">

<collection document-category="dftxml" usage="Data" name="default"/>

...

</domain>

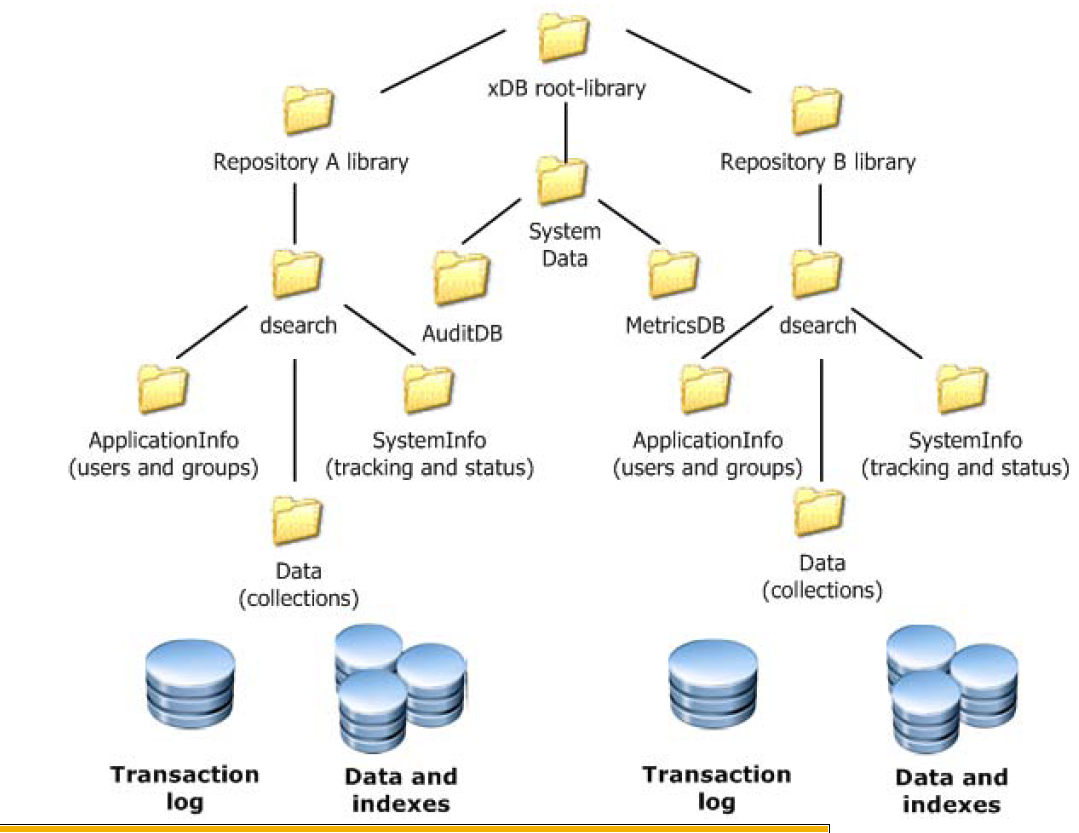
***Documentum categories***

A document category defines the characteristics of XML documents that belong to that category and their processing. All documents are sent to a specific index based on the document category. For example, xPlore pre-defines a category called dftxml that defines the indexes. All Documentum indexable content and metadata are sent to this category.

The following Documentum categories are defined within the *domain* element in indexserverconfig.xml.

* dftxml: XML representation of object metadata and content for full text indexing. To view the dftxml representation using xPlore administrator, click the document in the collection view.
* acl: ACLs that defined in the repository are indexed so that security can be evaluated in the full-text engine.
* group: Groups defined in the repository are indexed to evaluate security in the full-text engine.

***Mapping of domains to xDB***



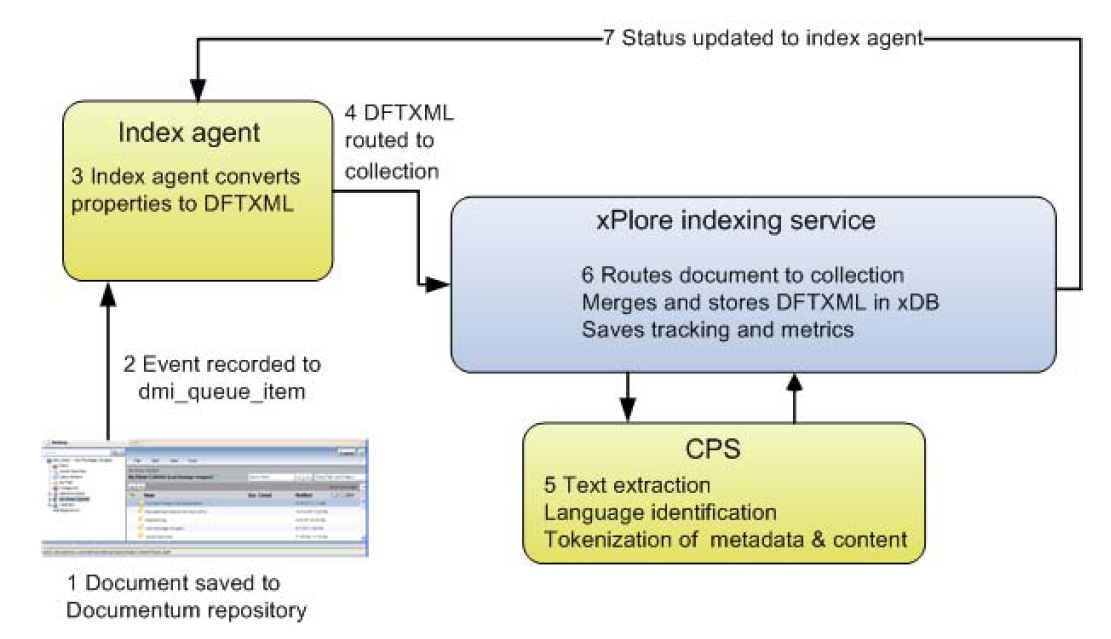
• The entire xPlore federation library is stored in xDB root-library.

• One content source (Documentum repository A) is mapped to a domain library. The library isstored in a defined storage area on either instance.

• A second repository, Repository B, has its own domain

* All xPlore domains share the system metrics and audit databases (SystemData library in xDB with libraries MetricsDB and AuditDB). The metrics and audit databases have a subcollection for each xPlore instance.
* The ApplicationInfo library contains Documentum ACL and group collections for a specific domain (repository).
* The SystemInfo library has two sub-collections: TrackingDB and StatusDB. Each collection in TrackingDB matches a collection in Data and is bound to the same instance as that data collection. There is a subcollection in StatusDB for each xPlore instance. The instance-specific sub-collection has a file *status.xml* that contains processing information for objects processed by the instance.
* The Data collection has a default sub-collection.

**How Content Server documents are indexed**



1. In a client application, a Save, Checkin, Destroy, Readonlysave, or MoveContent operation is

performed on a SysObject in the repository.

2. This operation event generates a queue item (*dmi\_queue\_item*) in the repository that is sent to the full-text user work queue. (The full-text user, *dm\_fulltext\_index\_user*, is a Superuser created when a repository is created or when an existing repository is upgraded.) The index agent retrieves the queue item and applies index agent filters. After an index request is submitted to xPlore, the client application can move on to the next task. (Indexing is asynchronous.)

3. The index agent retrieves the object associated with the queue item from the repository. The content is retrieved or staged to a temporary area. The agent then creates a *dftxml* (XML) representation of the object that can be used full-text and metadata indexing.

4. The Index Agent sends the dftxml representation of the content and metadata to the xPlore Server.

5. The xPlore indexing service calls CPS to perform text extraction, language identification, and

transformation of metadata and content into indexable tokens.

6. The xPlore indexing service performs the following steps:

* Routes documents to their target collections.
* Merges the extracted content into the dftxml representation of the document.
* Calls xDB to store the dftxml in xDB.
* Returns the tokens from CPS to xDB.
* Stores the document location (collection) and document ID in the TrackingDB.
* Saves indexing metrics in the MetricsDB.
* Tracks document indexing status in the StatusDB.

7. The indexing service notifies the index agent of the indexing status. The index agent then removes the queue item from the Content Server. Otherwise, the queue item is left behind with the error status and error message.

The object is now searchable

**How Content Server documents are queried**

Several software components control full-text search using the xPlore server:

* The Content Server queries the full-text indexes and returns query results to client applications.
* The xPlore server responds to full-text queries from Content Server.

***Path of a query from a Documentum client to xPlore***

1. The client application submits a DQL query or XQuery to the Documentum Content Server. (If

the client application uses DFC 6.6 or higher to create the query, DFC translates the query into

XQuery syntax.)

2. The Server transmits the query to xPlore (Content Server 6.6 or higher). The query plugin translates the query into XQuery syntax.

3. The query plugin transmits batches of HTTP messages containing XQuery statements to the xPlore search service.

4. CPS tokenizes the query based on the locale declared in the query. xDB breaks the query into XQuery clauses for full-text (using *ftcontains*) and metadata (using value constraints). The query is executed in the Lucene index against all collections unless a collection is specified in the query.

5. xDB applies the xPlore security filter to evaluate the security of the search results. If Documentum security evaluation is enabled, then security evaluation is done by the Content Server.

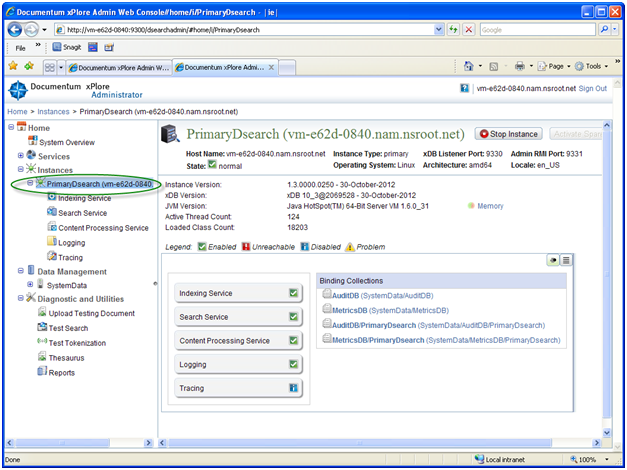
6. The results are returned in batches, with summary and facets.

## Managing the System

1. xPlore Admin Portal: Xplore Primary Instance)

http://host:9300/dsearchadmin

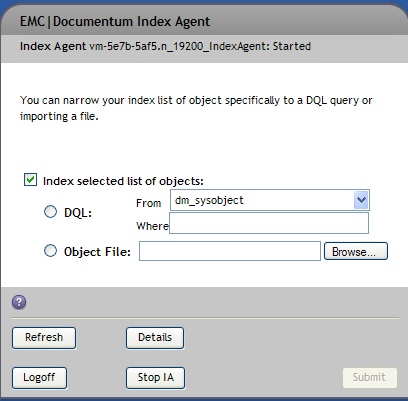
The xPlore administrator home page displays a navigation tree in the left pane and links to the four management areas in the content pane.



2. xPlore IndexAgent (IA) console:

<http://host:n9200/IndexAgent/login_dss.jsp>

For the first Index Agent you install, the http port is 19200, for the secnd it is 29200 on so on.



xPlore installation can have more than one IndexAgent installed and configured, covering different docbases.

However, each docbase can have only one IA configured on any given xPlore Host.

Each IA has a different TCP/IP port assigned to it. By convention, the fist IA you configure, is assigned port 19200. For each subsequent IA added, this port number in incremented by 10000.

**Starting and stopping the system**

1. Start or stop the primary instance using the start or stop scripts in *xplore\_home*/jboss5.1.0/server

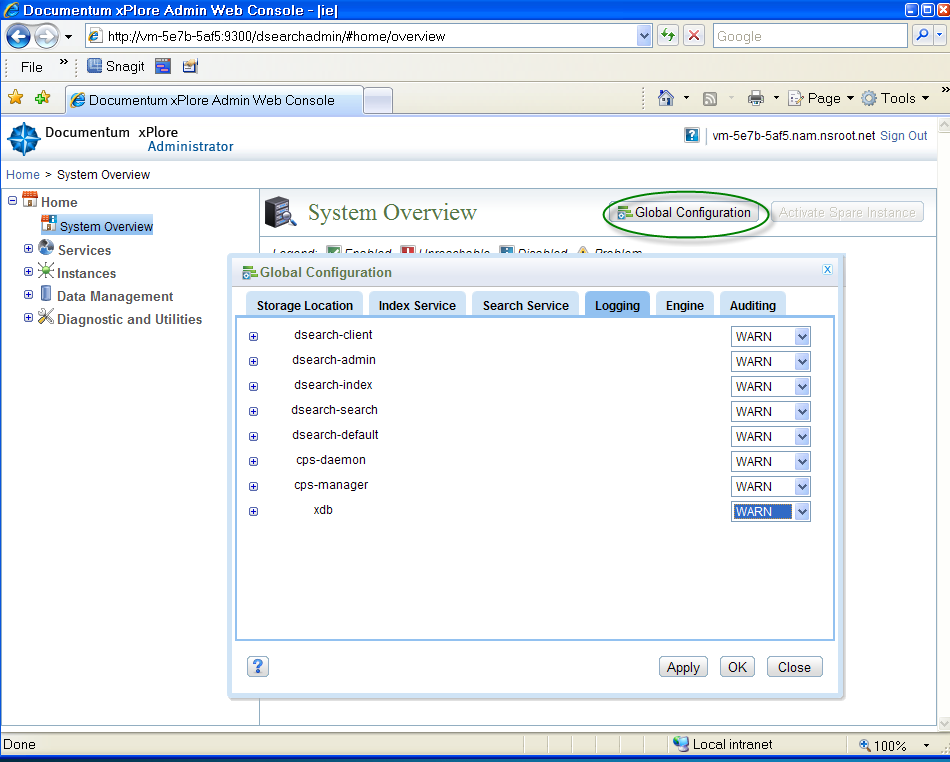
* /home/<instllation\_owner>/xPlore/jboss5.1.0/servers / startPrimaryDsearch.sh
* /home/<instllation\_owner>/xPlore/jboss5.1.0/servers / stopPrimaryDsearch.sh

1. Start or stop Index Agents using the start or stop script in *xplore\_home*/jboss5.1.0/server

* /home/<instllation\_owner>/xPlore/jboss5.1.0/servers/start<docnase>IA.sh
* /home/<instllation\_owner>/xPlore/jboss5.1.0/servers/stop<docnase>IA.sh

**Logging**

Logging levels are set in *Global Configuration* Panel of xPlore Admin Portal (http://host:9300/dsearchadmin)



Location of various log files is as follows:

**Search**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/logs/ dsearch.log

**CPS**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/logs/ cps\_daemon.log

**xDB**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/logs/ xdb.log

Index: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_<docbase>IA/logs/ <docbase>IA.log

**xPlore Instance**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/log/server.log

# Installation anD configuration

There are 4 hosts to configure as part of xPlore HA setup:

2 Content servers : CS-A and CS-B RHEL 6.x

2 xPlore servers : xPl-A and xPl-B. RHEL 6.x

## Configuring xPlore instance

All xPlore hosts have to be installed identically: same ownerships, same dir structure, same password

**Host environment setup: perform these steps on each xPlore host. You have to run them as root**

1. Configure swap-space:
2. Check if /swapfile exists**: ls –latr /swapfile**
3. Run**: /sbin/swapon -s**

If swap space is configured, you will see something, like:

Filename Type Size Used Priority

/swapfile file 4346872 2717380 -1

1. If swap space is not configured, run following sequence:

**dd if=/dev/zero of=/swapfile bs=1M count=4245**

**/sbin/mkswap /swapfile**

**/sbin/swapon /swapfile**

**/sbin/swapon –s**

You should see:

Filename Type Size Used Priority

/swapfile file 4346872 121288 -1

1. Update /etc/fstab, append following line:

/swapfile swap swap defaults 0 0

1. Configure NAS mount
2. Check if NAS share is mounted: **df –k**
3. If not mounted, mount it as a root on /nas folder.

mount filer:/dir /nas

1. Update /etc/fstab, append following

filer:/dir /nas nfs rw,hard,intr 0 0

**XPlore instance Configuration:**

After MST has successfully installed xPlore stack, the primary DSearch instance is configured and running.

For each of xPl-A and xPl-B do following:

### Finalize Primary Instance configuration

**Edit /home/dmadmin/ config/indexserverconfig.xml file. Use vi.**

* 1. Add the following properties under <category name …> tags:

<category name="dftxml">

<properties>

<property value="en" name="index-default-locale"/>

<property value="en" name="index-metadata-default-locale"/>

<property value="true" name="track-location"/>

<property value="true" name="enable-freshness-score"/>

</properties>

<category name="acl">

<properties>

<property value="false" name="track-location"/>

<property value="en" name="index-default-locale"/>

</properties>

<indexes>

<path-value-index options="CONCURRENT" path="/acl[object\_name&lt;STRING&gt; + owner\_name&lt;STRING&gt; + accessors/accessor[r\_accessor\_permit&lt;INT&gt; + r\_permit\_type&lt;INT&gt; + r\_accessor\_name&lt;STRING&gt; + r\_is\_group&lt;STRING&gt;]]" name="acl"/>

<path-value-index options="" path="/acl[object\_name&lt;STRING&gt; + owner\_name&lt;STRING&gt; + accessors/accessor[r\_accessor\_permit&lt;INT&gt; + r\_permit\_type&lt;INT&gt; + r\_accessor\_name&lt;STRING&gt; + r\_is\_group&lt;STRING&gt;]]" name="acl"/>

<path-value-index options="" path="/acl[r\_object\_id&lt;STRING&gt;]" name="aclobjectid"/>

</indexes>

</category>

<category name="group">

<properties>

<property value="false" name="track-location"/>

<property value="en" name="index-default-locale"/>

</properties>

<indexes>

<path-value-index options="CONCURRENT" path="/group[group\_name&lt;STRING&gt;]" name="group"/>

<path-value-index options="" path="/group[group\_name&lt;STRING&gt;]" name="group"/>

<path-value-index options="" path="/group[r\_object\_id&lt;STRING&gt;]" name="groupobjectid"/>

</indexes>

</category>

* 1. Add following <backup-location path> property:

<backup-location path="/nas/xPlore/backup/xPl-A"/>

* 1. Bump up *index-server-configuration* revision, like:

<index-server-configuration revision="1.15" ….

Will become:

<index-server-configuration revision="1.16" ….

**Edit /home/dmadmin/xPlore/dsearch/cps/cps\_daemon/PrimaryDsearch\_local\_configuration.xml:**

<request\_time\_out>1200</request\_time\_out>

**Edit /home/dmadmin/xPlore/watchdog/config/dsearch-watchdog-config.xml**

<property name="mail\_sender" value="Use email address specified at MST provisioning time"/>

<recurrence time-unit="minutes" frequency="8"/> /\* Multiple occurrences of this tag, all to be changed

### Restart xPlore instance

**$ cd /home/dmadmin/xPlore/jboss5.1.0/server**

**$ ./stopPrimaryDsearch.sh**

**Wait a while and check if all xPlore processes are gone.**

**$ ps –ef | grep xPlore**

**//Startup**

**$ nohup ./startPrimaryDsearch.sh &**

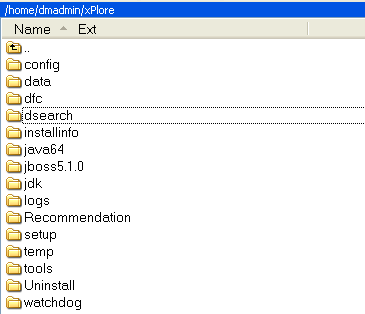
1. Verify xPlore

Login as dmadmin:

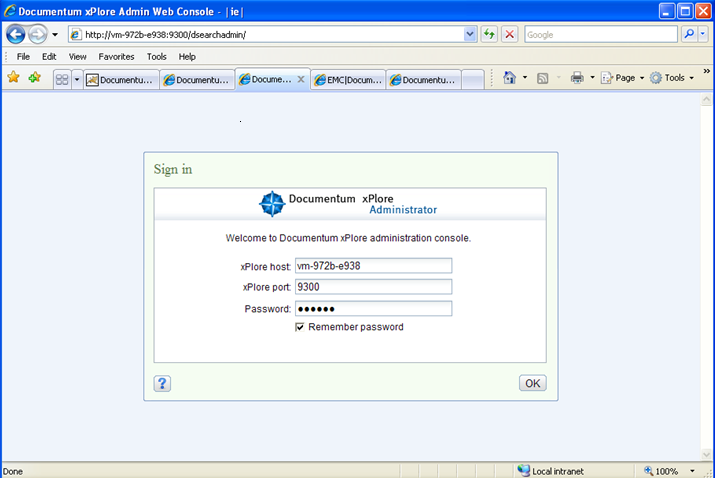
* 1. ps –ef | grep Dsearch, you should see something like:

dmadmin 19822 19706 0 Feb10 pts/1 00:12:09 /home/dmadmin/xPlore/java64/1.6.0\_31/bin/java -Dprogram.name=run.sh -server -Xms256m -Xmx4096m -XX:PermSize=64m -XX:MaxPermSize=256m -Xss1024k -XX:+UseCompressedOops -XX:+DisableExplicitGC -XX:-ReduceInitialCardMarks -Djava.awt.headless=true -Dorg.jboss.resolver.warning=true -Dsun.rmi.dgc.client.gcInterval=3600000 -Dsun.rmi.dgc.server.gcInterval=3600000 -Djava.endorsed.dirs=/home/dmadmin/xPlore/jboss5.1.0/lib/endorsed -classpath /home/dmadmin/xPlore/jboss5.1.0/bin/run.jar:/home/dmadmin/xPlore/java64/1.6.0\_31/lib/tools.jar org.jboss.Main -c DctmServer\_PrimaryDsearch -b 0.0.0.0

* 1. xPlore home dir (home/dmadmin/xPlore) should contain following folders:

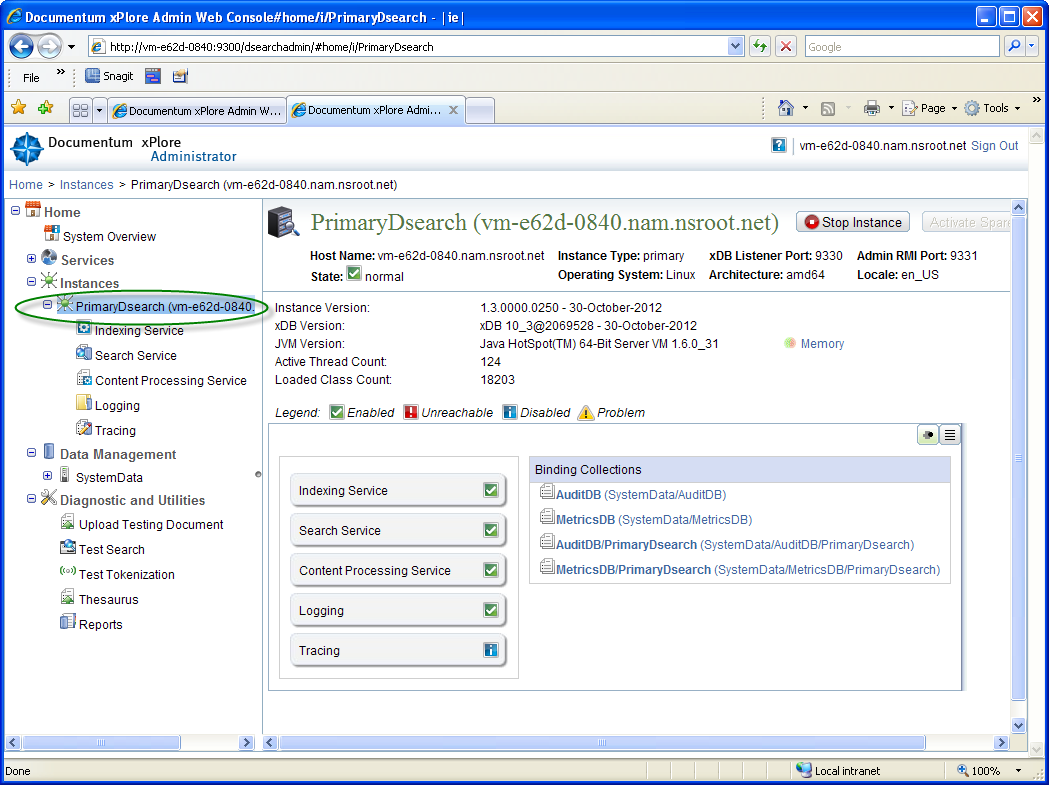


* 1. Go to following URL: http:// xPl-A(B):9300/dsearchadmin/



Enter the password you specified during MST provisioning, check the Remember Password checkbox and press OK

The UI page will come up: Click on ‘System Overview’ and you should see a single xPlore instance. Make sure all Services except Tracing are up (Green boxes are checked).



* 1. Run: top –U dmadmin. You should see following:

---------------------------------

top - 19:28:46 up 8 days, 5:42, 4 users, load average: 1.08, 1.02, 1.01

Tasks: 166 total, 3 running, 163 sleeping, 0 stopped, 0 zombie

Cpu(s): 6.7%us, 18.6%sy, 0.0%ni, 74.6%id, 0.1%wa, 0.0%hi, 0.0%si, 0.0%st

Mem: 8061864k total, 5871280k used, 2190584k free, 81148k buffers

Swap: 4346872k total, 121316k used, 4225556k free, 885824k cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND

19774 dmadmin 20 0 7548m 1.7g 10m S 0.7 22.3 10:40.62 java

19695 dmadmin 20 0 103m 1216 1056 S 0.0 0.0 0:00.00 startPrimaryDse

20318 dmadmin 20 0 822m 65m 15m S 0.0 0.8 0:10.25 CPSDaemon

20375 dmadmin 20 0 736m 52m 15m S 0.0 0.7 0:10.12 CPSDaemon

Make sure there are 2 CPSDaemons, 1 startPrimaryDse and 1 java processes running.

Verify that swap-space and free memory are available (at least 500 mBytes)

3. Backup xPlore.

All xPlore FileSystem backup are stored on NAS mount at following locations:

/nas/xPlore/backup/<xPL hostname>

Stop xPlore Primary Instance

cd /home/dmadmin

tar -zcvf /nas/xPlore/backup/<xPl-A>/ xPlore-PI-configured.tar.Z xPlore

tar -zcvf /nas/xPlore/backup/<xPl-A>/ xPlore-1.3-SilentInstaller.tar.Z xPlore-1.3-SilentInstaller

Restart xPlore Primary Instance

## Configure Index Agents.

Repeat steps in 3.2.1 and 3.2.2 each time you add a new Index Agent.

Note:

When configuring Index Agents you have to assign a specific HTTP port number to it.

The very first Index Agnet you configure is assigned port 19200. For each additional IA for a different docbase, increment the port number by 10000. Thus, the second IA, which have port 29200 and so on.

Always keep a table of currently installed Index Agents, like

|  |  |  |  |
| --- | --- | --- | --- |
| Domain/Broker | IA Name | IA Port | IA URI |
| trash on vm-c359-e6b9:1489 | vm-5e7b-5af5.n\_19200\_IndexAgent | 19200 | http://vm-5e7b-5af5.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent |
| trash on vm-297b-0827:1489 | vm-e62d-0840.n\_19200\_IndexAgent | 19200 | http://vm-e62d-0840.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent |
| testdoc on vm-c359-e6b9:1489 | vm-5e7b-5af5.n\_29200\_IndexAgent | 29200 | http://vm-5e7b-5af5.nam.nsroot.net:29200/IndexAgent/servlet/IndexAgent |
| testdoc vm-297b-0827:1489 | vm-e62d-0840.n\_29200\_IndexAgent | 29200 | http://vm-e62d-0840.nam.nsroot.net:29200/IndexAgent/servlet/IndexAgent |

## Configure xPl-A IA

1. Clean-up FT related objects in the docbase.Use IDQL to connect as super-user to the docbase for which you are configuring IA.

Run following DQL:

**select** **object\_name from dm\_ftengine\_config**

If object\_name is “FAST Fulltext Engine Configuration”, quit IDQL and run on CS-A:

cd /opt/dmadmin/CS67SP1/product/6.7SP1/install/admin

dmbasic -f create\_fulltext\_objects\_ha.ebs -e HACleanupBeforeUpgradeStep -- <docbase\_name> <SUPER-USER-ID> <PASSWORD>

Output will look lke:

Connected to docbase testdoc as user xxxxxxx.

Successfully deleted dm\_ftengine\_config objects: '0800270f80002148'

Successfully deleted dm\_fulltext\_index objects: '3b00270f80000100'

Found no dm\_fulltext\_collection objects to clean up.

If object\_name is empty, proceed to next step.

2. Run IA silent installer

**Edit /home/dmadmin/xPlore-1.3-SilentInstaller/config/silent.properties**

All configurable properties of Index Agents are specified in #=======configure the index agent==== section of this file.

This section consists of multiple numbered blocks of parameters for each Index Agent.

For example, the first IA (trash docbase) is defined by following block:

Notice that the first IA is assigned instance.port=19200. You have to increment this number by 10000 for each additional IA installed.

vm-5e7b-5af5.nam.nsroot.net is xPl-A

vm-c359-e6b9 is CS-A

/home/dmadmin/xPlore-1.3-SilentInstaller/config/silent.properties :

# trash IA

indexagent.instance.name.1=trashIA

indexagent.instance.fqdn.1=vm-5e7b-5af5.nam.nsroot.net

indexagent.instance.ip.1=

indexagent.instance.port.1=19200

indexagent.instance.password.1=qw12!@ / use same pass provuided during MST provisioning

indexagent.docbase.name.1=trash

indexagent.docbase.user.1=doctester

indexagent.docbase.password.1=Welcome1

**indexagent.connectionBroker.host.1=<CS-A Host> like vm-c359-e6b9**

indexagent.connectionBroker.port.1=1489

indexagent.globalRegistryRepository.name.1=globalstore

indexagent.globalRegistryRepository.user.1=dm\_bof\_registry

indexagent.globalRegistryRepository.password.1=bof123

The IA for testdoc docbase is defined by following block:

# testdoc IA

indexagent.instance.name.2=testdocIA

indexagent.instance.fqdn.2=vm-5e7b-5af5.nam.nsroot.net

indexagent.instance.ip.2=

indexagent.instance.port.2=29200

indexagent.instance.password.2=qw12!@

indexagent.docbase.name.2=testdoc

indexagent.docbase.user.2=doctester

indexagent.docbase.password.2=Welcome1

**indexagent.connectionBroker.host.2=<CS-A Host> like vm-c359-e6b9**

indexagent.connectionBroker.host.2=vm-c359-e6b9

indexagent.connectionBroker.port.2=1489

indexagent.globalRegistryRepository.name.2=globalstore

indexagent.globalRegistryRepository.user.2=dm\_bof\_registry

indexagent.globalRegistryRepository.password.2=bof123

**Run silent installer for every Agent defined in /home/dmadmin/xPlore-1.3-SilentInstaller/config/silent.properties.**

$ cd /home/dmadmin/xPlore-1.3-SilentInstaller

$ export JAVA\_HOME=./jdk

$ export ANT\_HOME=./ant

$ export PATH=${PATH}:${JAVA\_HOME}/bin:${ANT\_HOME}/bin

// If this is the first IA being configured, start with Dindexagent.number=1

ant -f ./script/build.xml config.indexagent -Dindexagent.number=1 | tee config.<docbase>.indexagent.out

**3. Finalize IA configuration**

***Edit /home/dmadmin/xPlore/setup/indexagent/tools/ftintegrity\_for\_ <docbase>.sh.***

Edit this script (changes highlighted) , and specify the docbase user/password:

$JAVA\_HOME/bin/java -classpath "./lib/stateofindex.jar:$CONFIG\_DIR:$DOCUMENTUM/dfc/log4j.jar:./lib/httpcore-4.0.1.jar:./lib/httpclient-4.0.1.jar:./lib/commons-logging-1.1.1.jar:./lib/commons-codec-1.3.jar:./lib/commons-pool-1.6.jar:$DOCUMENTUM/dfc/dfc.jar:$DOCUMENTUM/dfc/aspectjrt.jar:./lib/dsearch-client.jar:./lib/dsearch-utils.jar:./lib/dsearch-install.jar:./lib/slf4j-api-1.6.4.jar:$DOCUMENTUM/dfc/activation.jar;$DOCUMENTUM/dfc/jaxb-api.jar:$DOCUMENTUM/dfc/jaxb-impl.jar:$DOCUMENTUM/dfc/jsr173\_api.jar" com.documentum.server.impl.method.fulltext.stateofindex.FTStateOfIndex trash doctester Welcome1 vm-5e7b-5af5.nam.nsroot.net 9300 $DSEARCH\_DOMAIN $OUTPUT\_FILEPATH $BATCH\_SIZE –useFilter T

***Edit /home/dmadmin/xPlore/setup/indexagent/tools/aclreplication\_for\_ <docbase>.sh.***

Edit this script (changes highlighted) , and specify the docbase user/password:

$JAVA\_HOME/bin/java -classpath "./lib/aclreplication.jar:$CONFIG\_DIR:$DOCUMENTUM/dfc/log4j.jar:./lib/server-impl.jar:./lib/httpclient-4.0.1.jar:./lib/httpcore-4.0.1.jar:./lib/commons-logging-1.1.1.jar:./lib/commons-codec-1.3.jar:./lib/commons-pool-1.6.jar:./lib/aclxmlgen.jar:$DOCUMENTUM/dfc/dfc.jar:$DOCUMENTUM/dfc/aspectjrt.jar:./lib/dsearch-client.jar:./lib/dsearch-utils.jar:./lib/dsearch-install.jar:./lib/slf4j-api-1.6.4.jar:$DOCUMENTUM/dfc/activation.jar;$DOCUMENTUM/dfc/jaxb-api.jar:$DOCUMENTUM/dfc/jaxb-impl.jar:$DOCUMENTUM/dfc/jsr173\_api.jar" com.documentum.server.impl.method.fulltext.aclreplication.FTIndexACLGroups testdoc doctester Welcome1 vm-5e7b-5af5.nam.nsroot.net 9300 $DSEARCH\_DOMAIN

***Edit /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_trashIA/deploy/IndexAgent.war/WEB-INF/classes/indexagent.xml.***

Insert following section under <index-agent-instance> element at the top of this xml:

<parameter\_list>

<parameter>

<parameter\_name>type\_based\_migration</parameter\_name>

<parameter\_value>dm\_sysobject</parameter\_value>

</parameter>

</parameter\_list>

Change following values:

<content\_clean\_interval>2400000</content\_clean\_interval>

<runaway\_item\_timeout>2400000</runaway\_item\_timeout>

<parameter\_list>

<parameter>

<parameter\_name>contentSizeLimit</parameter\_name>

<parameter\_value>40000000</parameter\_value>

</parameter>

</parameter\_list>

<local\_content\_area>/home/dmadmin/xPlore/data/<docbase>IA/export

### 4. Restart xPlore instance and Index Agent on xPl-A

**$ cd /home/dmadmin/xPlore/jboss5.1.0/server**

**$ ./stopPrimaryDsearch.sh**

**$ ./stop<docbase>IA.sh**

**Wait a while and check if all xPlore processes are gone.**

**$ ps –ef | grep xPlore**

**//Startup**

**$ nohup ./startPrimaryDsearch.sh &**

**$ nohup ./start<docbase>IA.sh &**

### At this point, Primary DSearch and configured Index Agent is running. Visit URL for DSearch Admin console and each IA console and verify by logging in IA Console UI)

<http://vm-5e7b-5af5:9300/dsearchadmin>

<http://vm-5e7b-5af5:[1-N]9200/IndexAgent/login_dss.jsp>

### 5. Run DB consistency check on Data Management node of dsearchadmin Console.

## Configure IA on xPl-B

### Note: installing IA on xPL-B host is similar, but not identical to installing IA on xPl-B.

### 1. On CS-A run:

cd /opt/dmadmin/CS67SP1/product/6.7SP1/install/admin

dmbasic -f create\_fulltext\_objects\_ha.ebs -e HAPreInstallStep -- <docbase> dmadmin <password>

2. Run IA silent installer

**Edit /home/dmadmin/xPlore-1.3-SilentInstaller/config/silent.properties**

All IndexAgents are specified in #=======configure the index agent==== section of this file.

This section consists of multiple numbered blocks of parameters for each Index Agent.

For example, the first IA (trash docbase) is defined by following block:

Notice that the first IA is assigned instance.port=19200. You have to increment this number by 10000 for each additional IA installed.

vm-5e7b-5af5.nam.nsroot.net is xPl-A

vm-c359-e6b9 is CS-A

# trash IA

indexagent.instance.name.1=trashIA

indexagent.instance.fqdn.1=vm-5e7b-5af5.nam.nsroot.net

indexagent.instance.ip.1=

indexagent.instance.port.1=19200

indexagent.instance.password.1=qw12!@

indexagent.docbase.name.1=trash

indexagent.docbase.user.1=doctester

indexagent.docbase.password.1=Welcome1

**indexagent.connectionBroker.host.1=<CS-B Host> like vm-297b-0827**

indexagent.connectionBroker.port.1=1489

indexagent.globalRegistryRepository.name.1=globalstore

indexagent.globalRegistryRepository.user.1=dm\_bof\_registry

indexagent.globalRegistryRepository.password.1=bof123

The IA for testdoc docbase is defined by following block:

# testdoc IA

indexagent.instance.name.2=testdocIA

indexagent.instance.fqdn.2=vm-5e7b-5af5.nam.nsroot.net

indexagent.instance.ip.2=

indexagent.instance.port.2=29200

indexagent.instance.password.2=qw12!@

indexagent.docbase.name.2=testdoc

indexagent.docbase.user.2=doctester

indexagent.docbase.password.2=Welcome1

**indexagent.connectionBroker.host.2=<CS-B Host> like vm-297b-0827**

indexagent.connectionBroker.port.2=1489

indexagent.globalRegistryRepository.name.2=globalstore

indexagent.globalRegistryRepository.user.2=dm\_bof\_registry

indexagent.globalRegistryRepository.password.2=bof123

**Run silent installer for every Agent defined in /home/dmadmin/xPlore-1.3-SilentInstaller/config/silent.properties.**

$ cd /home/dmadmin/xPlore-1.3-SilentInstaller

$ export JAVA\_HOME=./jdk

$ export ANT\_HOME=./ant

$ export PATH=${PATH}:${JAVA\_HOME}/bin:${ANT\_HOME}/bin

ant -f ./script/build.xml config.indexagent -Dindexagent.number=1 | tee config.trash.indexagent.out

ant -f ./script/build.xml config.indexagent -Dindexagent.number=2 | tee config.testdoc.indexagent..out

3. Finalize IA configuration

**Edit /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_trashIA/deploy/IndexAgent.war/WEB-INF/classes/indexagent.xml.**

Insert following section under <node> element at the top of xml:

<parameter\_list>

<parameter>

<parameter\_name>type\_based\_migration</parameter\_name>

<parameter\_value>dm\_sysobject</parameter\_value>

</parameter>

</parameter\_list>

Change following values:

<content\_clean\_interval>2400000</content\_clean\_interval>

<runaway\_item\_timeout>2400000</runaway\_item\_timeout>

<parameter\_list>

<parameter>

<parameter\_name>contentSizeLimit</parameter\_name>

<parameter\_value>40000000</parameter\_value>

</parameter>

</parameter\_list>

<local\_content\_area>/home/dmadmin/xPlore/data/<docbase>IA/export

4. Restart xPlore instance and Index Agents on xPl-B

**$ cd /home/dmadmin/xPlore/jboss5.1.0/server**

**$ ./stopPrimaryDsearch.sh**

**$ ./stop<docbase>IA.sh**

**Wait a while and check if all xPlore processes are gone.**

**$ ps –ef | grep xPlore**

**//Startup**

**$ nohup ./startPrimaryDsearch.sh &**

**$ nohup ./start<docbase>IA.sh &**

5. At this point, Primary DSearch and aconfigured IndexAgents are running, but Index Agents are stopped. Visit URL for DSearch Admin console and each IA console and verify. Do not start Index agents in UI.

<http://vm-5e7b-5af5:9300/dsearchadmin>

<http://vm-5e7b-5af5:19200/IndexAgent/login_dss.jsp>

<http://vm-5e7b-5af5:29200/IndexAgent/login_dss.jsp>

<http://vm-5e7b-5af5:39200/IndexAgent/login_dss.jsp>

### 6. Update FT related object in docbase. On CS-A run:

### cd /opt/dmadmin/CS67SP1/product/6.7SP1/install/admin

// Run following for each configured docbase

dmbasic -f create\_fulltext\_objects\_ha.ebs -e HAPostInstallStep -- <docbase> dmadmin <password>

7. Update dm\_ftengine\_config objects (there are 2) in the Run following DQL:

IDQL> select r\_object\_id from dm\_ftengine\_config

This will retrun 2 ids

* 1. Run following IAPI sequence for each of these ids:

IAPI> append,c,<r\_object\_id>,param\_name

SET> fast\_wildcard\_compatible

IAPI> append,c,<r\_object\_id>,param\_value

SET> true

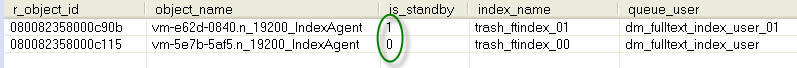
IAPI> save,c, <r\_object\_id>

8. Restart all effected docbases on both CS-A and CS-B.

9. Verify FT index configuration. Run following DQL in every docbase:

**select** ia.r\_object\_id,ia.object\_name,fi.is\_standby,ia.index\_name,ia.queue\_user **from** **dm\_ftindex\_agent\_config** ia,dm\_fulltext\_index fi **where** ia.index\_name=fi.index\_name

**Only one index agent should be on a standby (is\_standby=1)**



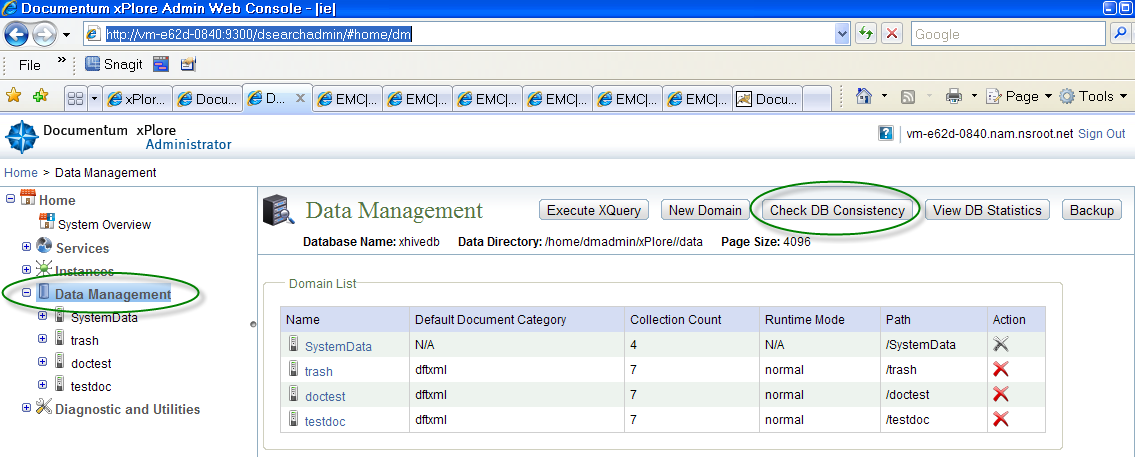
## Back-up xPlore.

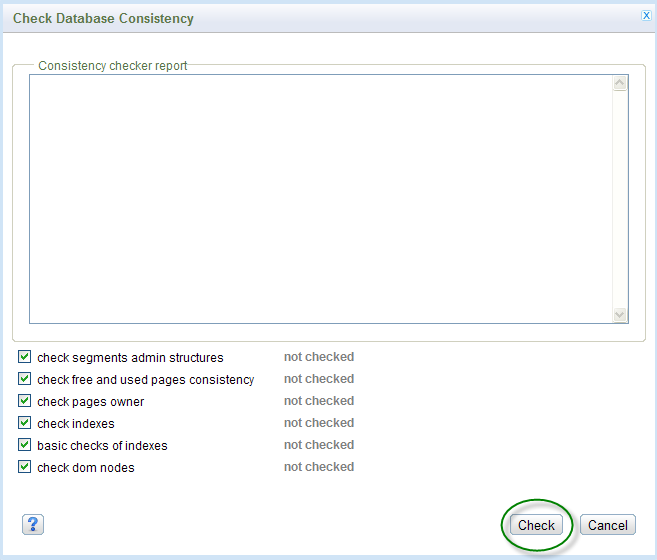
Verify integrity of xPlore installation on xPl-A and xPl-B and tarball both /home/dmadmin/xPLore folders.

Never backup xPlore which did not pass DB Consistency Check.

1. Check DB Consistency on Data Management Nodes using xPlore Admin console at

<http://xPloreHost:9300/dsearchadmin/>





1. Stop xPlore on both xPl-A and xPl-B
2. Tarball xPlore and Silent Installer folders on both xPl-A and xPl-B

On xPl-A:

cd /home/dmadmin

tar -zcvf /nas/xPlore/backup/<xPl-A>/xPlore-configured.tar.Z xPlore

tar -zcvf /nas2/xPlore/backup/xPl-A/ xPlore-1.3-SilentInstaller.tar.Z xPlore-1.3-SilentInstaller

On xPl-B:

cd /home/dmadmin

tar -zcvf /nas2/xPlore/backup/xPl-B/xPlore-configured.tar.Z xPlore

tar -zcvf /nas2/xPlore/backup/xPl-B/ xPlore-1.3-SilentInstaller.tar.Z xPlore-1.3-SilentInstaller

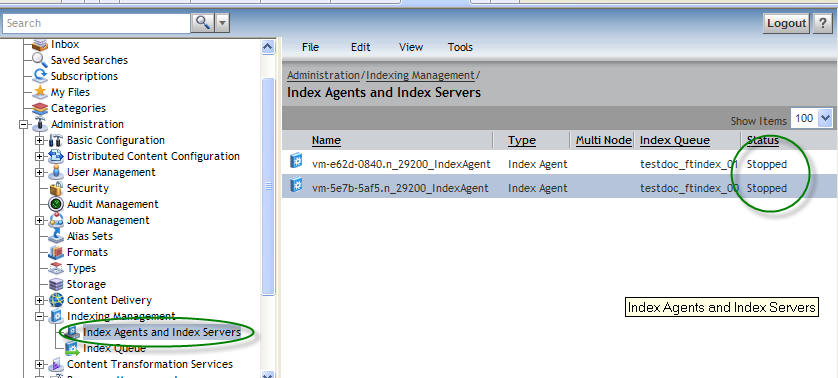
1. Restart xPlore on both xPl-A and xPl-B

# Initial indexing

At this point, both xPlore hosts are up and running, but Index Agents are stopped.

Steps to verify (repeat for each configured docbase):

1. Login to DA and go to Index Management Node | Index Agents and Index Servers:



Both Agents should be in Stopped Status.

Depending on the content size of the docbase, Initial re-indexing may take long time and mush resources (mostly CPU, memory, network and IO capacity.

To save time, we will re-index docbase content on xPl-A host and then copy index collections from xPl-A to xPl-B host.

## xPl-A

1. Verify xPlore processes are running on xPl-A host.

You shoud see (in this example, there are 3 Index Agent configured):

* Swap and free memory are available
* 2 CPS Daemons
* 4 java process (one for DSearch Admin and 3 Index Agents

$ top –U dmadmin

Mem: 8061844k total, 7572524k used, 489320k free, 133688k buffers

Swap: 4346872k total, 1049044k used, 3297828k free, 2491780k cached

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND

**31592 dmadmin 20 0 3810m 568m 11m S 1.0 7.2 10:01.43 java**

**31603 dmadmin 20 0 7403m 2.2g 11m S 1.0 28.9 15:22.30 java**

**31593 dmadmin 20 0 3746m 551m 11m S 0.7 7.0 10:04.42 java**

**31612 dmadmin 20 0 3746m 589m 11m S 0.7 7.5 9:57.40 java**

31996 dmadmin 20 0 822m 39m 3260 S 0.3 0.5 0:12.85 CPSDaemon

31482 dmadmin 20 0 103m 1052 1048 S 0.0 0.0 0:00.00 startPrimaryDse

31483 dmadmin 20 0 103m 1048 1044 S 0.0 0.0 0:00.00 starttrashIA.sh

31484 dmadmin 20 0 103m 1048 1044 S 0.0 0.0 0:00.00 starttestdocIA.

31485 dmadmin 20 0 103m 1048 1044 S 0.0 0.0 0:00.00 startdoctestIA.

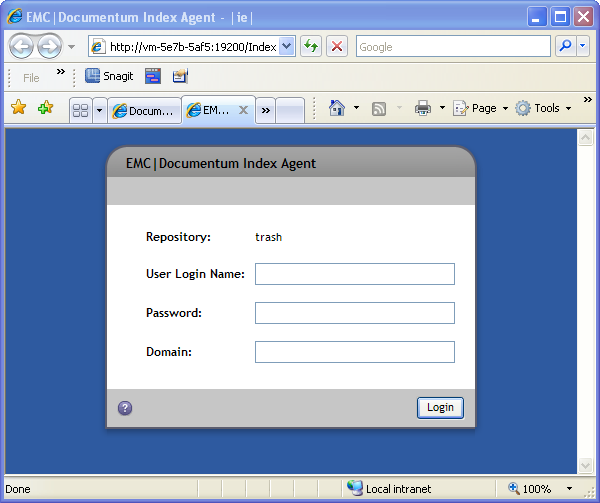
32115 dmadmin 20 0 736m 37m 3224 S 0.0 0.5 0:12.64 CPSDaemon

1. Login to Index Agent UI (each in a separate tab of IE:

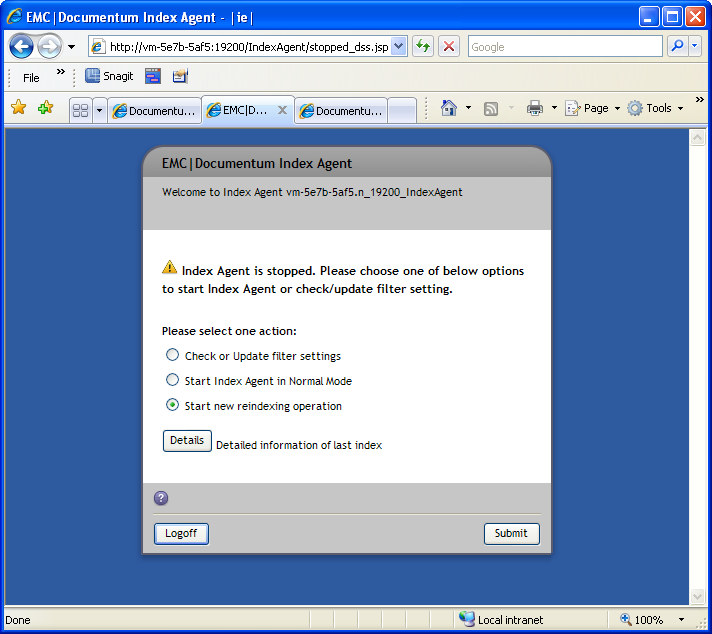
[http://xPl-A:19200/IndexAgent/login\_dss.jsp](http://xPL-A:19200/IndexAgent/login_dss.jsp)

[http://xPl-A:29200/IndexAgent/login\_dss.jsp](http://xPL-A:29200/IndexAgent/login_dss.jsp)

<http://xPl-A:39200/IndexAgent/login_dss.jsp>



1. For each configured docbase (start from -0RT 19200), do following:
   1. In IA console UI, Select ‘**Start** New Re-indexing Operation’ and submit



* 1. After submitting, you can follow the progress of re-indexing by ‘Refresh’ on following:

**Total count**: This is a count of dm\_sysobjects whch ae going to be indexed

**Processed Count**: Number of documents processed

**Filtered Count**: Some dm\_sysobjects are excluded from indexing by: for example, content of /System cabinet. The filter is configurable (by selecting ‘Check or Update filer settings’ on Index Agent UI); we are going to use default setting

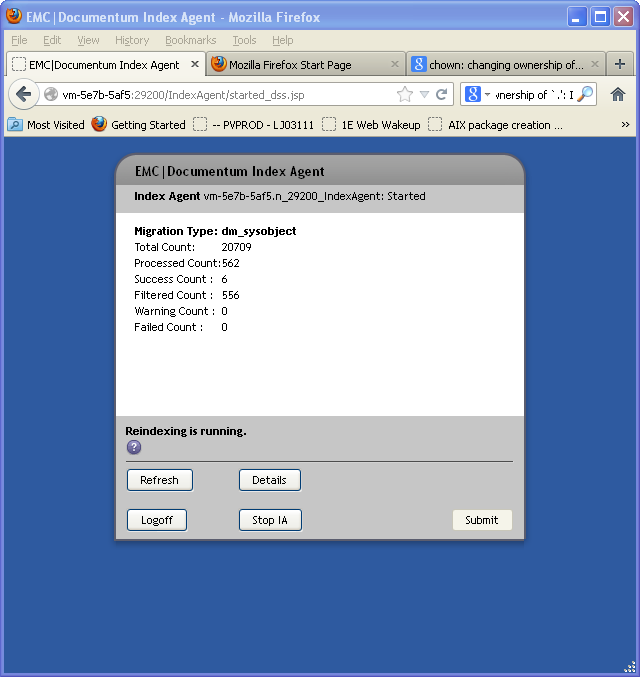
**Success Count**: Number of documents indexed

**Warning Count**: It is normal to see warning count to occasionally go up as indexing progresses. If the number of warnings keeps growing rapidly, ‘Stop Agent’ in Index Agent UI and check Index Agent log (grep on “WARN").

/home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_<docbase>IA/logs/\_<docbase>IA.log

After fixing problemS with excessive Warning Count, restart interrupted indexing by selecting ‘Continue Reindexing Operation’ on Index Agnet UI.

**Failed Count: See Warning Count…**

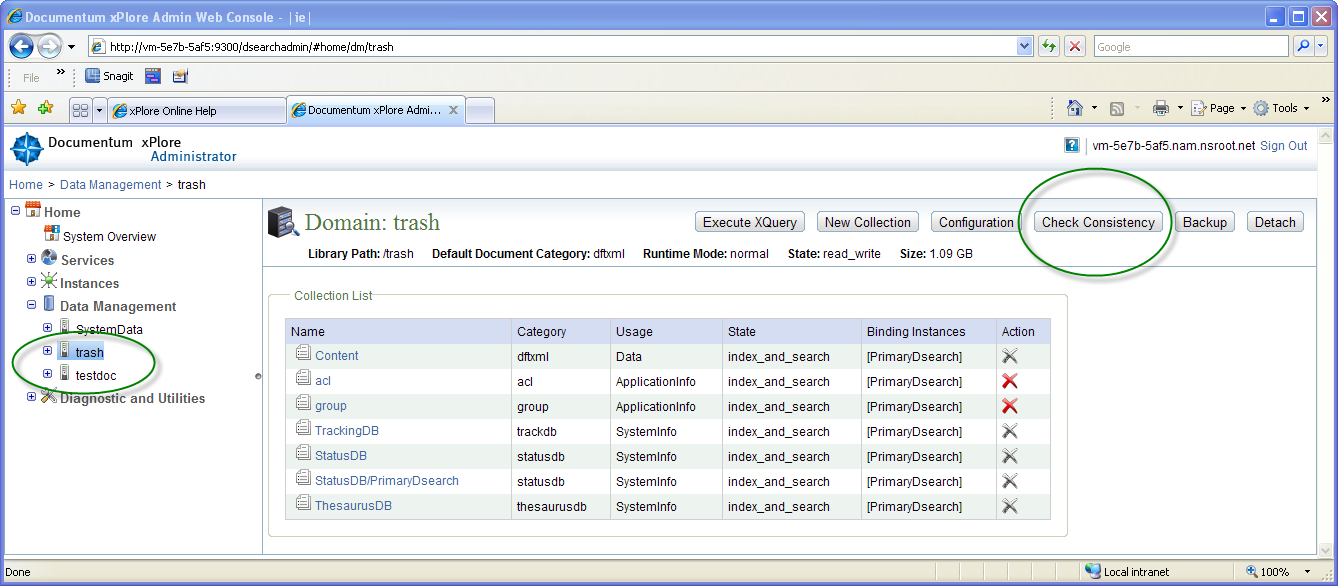


1. After re-indexing finishes, Stop all IA, And restart IA in a normal mode.
2. Synchronize ACL and Groups Collection
   1. cd /home/dmadmin/xPlore/setup/indexagent/tools
   2. Update aclreplication scripts to set docbase owner’s password. Owner id is same as specified in silent.properties:indexagent.docbase.user.n. Use the same password provided in matching silent.properties:indexagent.docbase. password .n

sed -i "s/<password>/Welcome1/g" aclreplication\_for\_<docbase>.sh

* 1. run: ./aclreplication\_for\_<docbase>.sh. Repeat for all configured docbases.

1. Run Consistency Check on ‘Data Management’ node in Dsearch admin console;

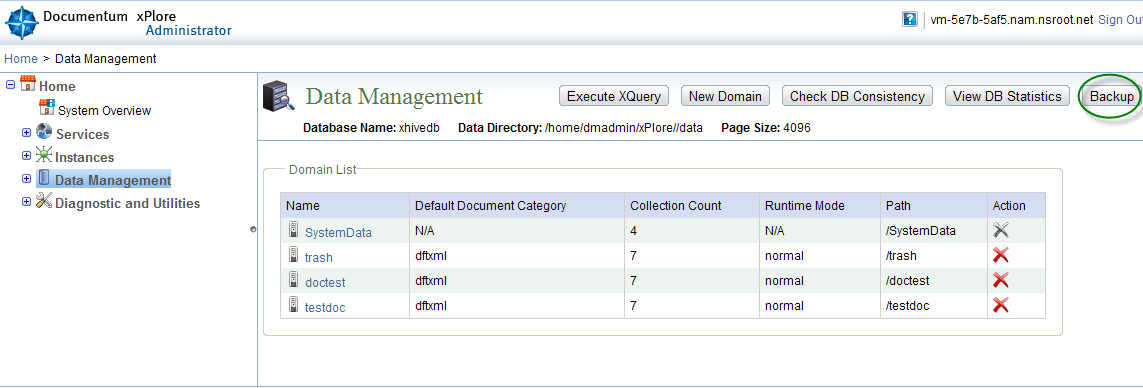


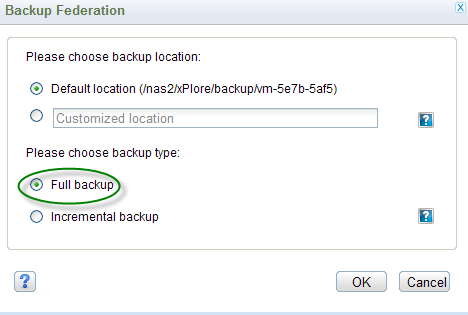
1. Do full backup of xPlore (xPl-A) Folder.

cd /home/dmadmin

tar -zcvf /nas/xPlore/backup/<xPl-A>/xPlore-indexed.tar.Z xPlore

1. Back-up xPlore federation.





1. Shut-down xPlore on xPl-A. Record the time the index agent is shut down.

## xPl-B

1. Shut-down xPlore on xPl-B.
2. Backup /home/dmadmin/xPlore/config/indexserverconfig.xml and /home/dmadmin/xPlore /config/XhiveDatabase.bootstrap on xPl-B. You will use this backup to recover configuration information after you transfer data from xPl-A.
3. Rename data and config folder

mv home/dmadmin/xPlore/data home/dmadmin/xPlore/data-prev

mv home/dmadmin/xPlore/config home/dmadmin/xPlore/config-prev

1. Copy the directories /home/dmadmin/xPlore /data and /home/dmadmin/xPlore /config from xPl-A to xPl-B. Do not change file attributes (cp –ra).
2. We are done transferring xPloe data files from xPl-A. Start xPlore and the index agents in normal mode (not migration mode) on xPl-A (not xPl-B).
3. Update xplore\_home/config/indexserverconfig.xml on xPl-B with settings in the original backup copy. Change the attributes “url” and “hostname” of the XML element index-server-configuration.node to match those values in the backup file. Bump up xml revision, like like:

<index-server-configuration revision="1.40"

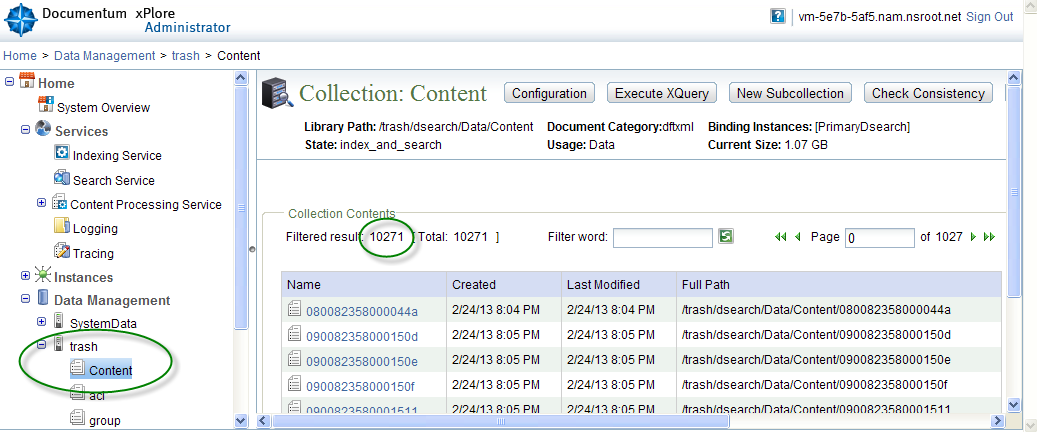
To:

<index-server-configuration revision="1.41"

1. Update xplore\_home/config/XhiveDatabase.bootstrap on xPl-B to match the original file. Change attribute “host” of the element /server/node to match the value in the backup file.
2. Start xPlore and the index agent in normal mode on xPl-B.
3. Run aclreplication script to index ACLs and DM\_GROUP objects.

Note: Steps are different from those used to run ACL replication on xPl-A

1. Swap standny ft\_engine.
2. Run aclreplication
3. Login to dsearchadmin consoles on both xPl-A and xPl-B and run ‘Check DB Consintency’
4. Drill down to all indexed domains/collections and compare total index count.



# Maintainance

Most system administration tasks are available in xPlore administrator. When you open xPlore administrator, you see the navigation tree and the system overview page. From the tree, you can open administration pages for system-wide services, instance-specific services, data management, and diagnostics and troubleshooting. From the system overview, click the grid symbol to see the following information for each xPlore instance: component status, instance description, instance status, JVM information, and runtime information.

When you open a service page, such as indexing service, the actions apply to all indexing services in the xPlore installation. To change the indexing service configuration for a specific instance, open the instance in the navigation tree and then choose the service.

## Enabling indexing for an object type

Events in dmi\_registry for the user *dm\_fulltext\_index\_user* generate queue items for indexing. The following events are registered for *dm\_fulltext\_index\_user* to generate indexing events by default:

* dm\_sysobject: dm\_save, dm\_checkin, dm\_destroy, dm\_saveasnew, dm\_move\_content
* dm\_acl: dm\_save, dm\_destroy, dm\_saveasnew
* dm\_group: dm\_save, dm\_destroy

## Registering a type for full-text indexing

Use Documentum Administrator to change the full-text registration for an object type. Select the type, view the properties, and for the property **Enable indexing** check **Register for indexing**. To change specific events that are registered for full-text, use the DFC API registerEvent().

## Reindexing

The index agent does not recreate all the queue items for reindexing. Instead, it creates a watermark queue item (type *dm\_ftwatermark*) to indicate the progress of reindexing. It picks up all the objects for indexing in batches by running a query. The index agent updates the watermark as it completes each batch. When the reindexing is completed, the watermark queue item is updated to ’done’ status.

You can submit for reindexing one or all documents that failed indexing. In Documentum

Administrator, open **Indexing Management > Index Queue**. Choose **Tools > Resubmit all failed queue items**, or select a queue item and choose **Tools > Resubmit queue item**.

## Monitoring xPlore

## Watchdog service

The watchdog service pings all xPlore instances and sends an email notification when an instance does not respond. To configure watchdog timing, edit the configuration file dsearch-watchdog-config.xml. This file is located in xplore\_home/watchdog/config. The following timing properties within the element timing-in can be configured:

* recurrence timeunit and frequency: Specifies how often the task is executed. For example, the disk space task with a frequency of 2 and time unit of hours checks disk space every two hours. Default: Every minute.
* start-date: date and time the task should be invoked, in UTC format. If the date is in the past, the task will be executed as soon as possible.
* expiry-date: Specifies the date and time a task stops executing, in UTC format.
* max-response-timeout: Specifies how long between detection of a hung task and execution of the notification (or other task). For example, a wait-time value of 6 and time unit of hours indicates a wait of 6 hours before notification about a non-responding instance.
* max-retry-threshold: Specifies the maximum number of times the task can be retried. For example, if the task is notification, a value of 10 indicates the notification task is retried 10 times. Recurring tasks are retried at the next scheduled invocation time.
* max-iterations: Maximum number of times to attempt to ping an instance that has no response. Default: -1 (no limit)

You can also configure the timing properties for the index agent.

The watchdog can detect out of space issues in xPlore instances. The watchdog monitors storage locations and the config and log directories. When the limit of available space is reached, the watchdog sends a notification to the administrator.

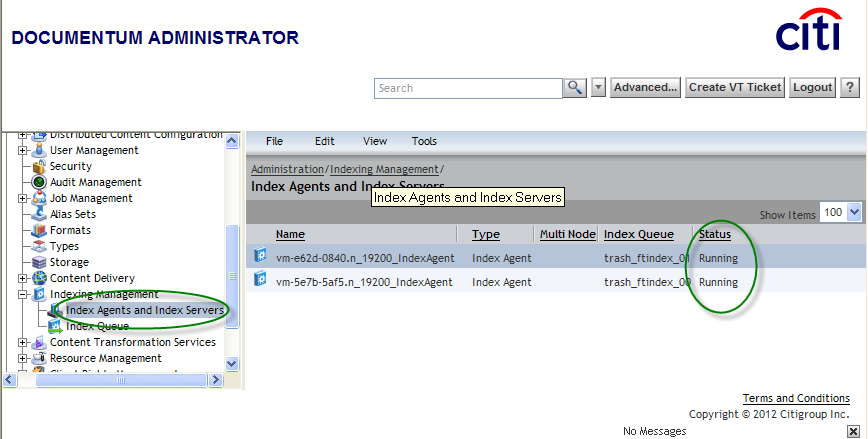
To configure watchdog disk space monitoring, edit the configuration file dsearch-watchdog-config.xml. This file is located in xplore\_home/watchdog/config. The task id is similar to PrimaryDsearch\_DiskFreeSpaceMonitor. The ID changes depending on the ID of the xPlore instance. The following properties can be configured. All but the first property are configured within the timing-info element.

* percent\_available\_space\_to\_take\_action: Value attribute: Percentage of free disk space. Default: 30.
* recurrence timeunit and frequency: Specifies how often disk space is checked. For example, a frequency of 2 and time unit of hours checks every two hours. Default: 120 minutes (2 hours).
* start-date: date and time the task should be invoked, in UTC format. If the date is in the past, the task will be executed as soon as possible.
* expiry-date: Specifies the date and time a task stops executing, in UTC format.
* max-response-timeout: Specifies how long between detection of a hung task and execution of the notification (or other task). For example, a wait-time value of 6 and time unit of hours indicates a wait of 6 hours before notification about a non-responding instance.
* max-retry-threshold: Specifies the maximum number of times the task can be retried before the next scheduled invocation time.

## DA

### Index Agent.

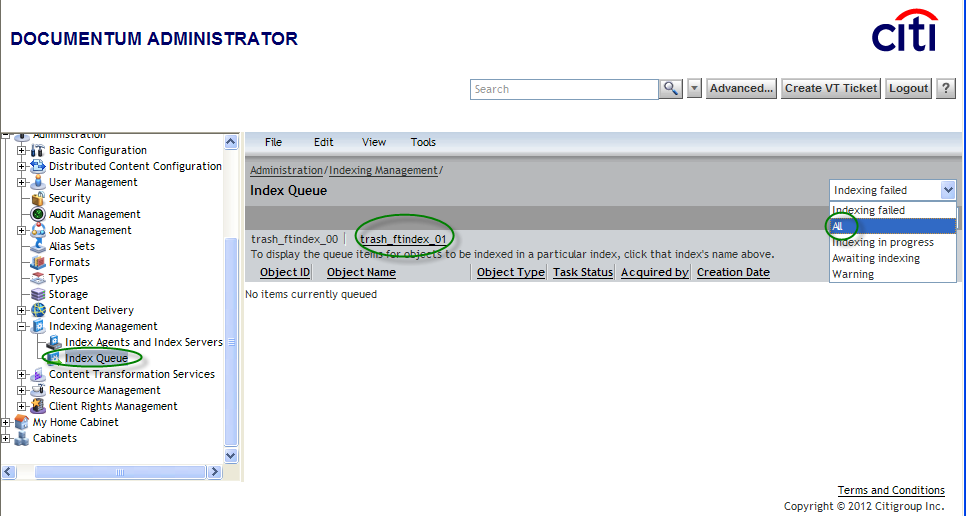
### Go through all docbases which are configured for Full Text Indexing:



Note, there are two Index Agents running on xPl-A and xPl-B. Both should have ‘Running’ status. To change the Status, right click on Agent name and change the state to Running.

**Index Queue.**

Each time a new content is created in the docbase or condent is updated, this document is put on ContentServer queue. Since in HA xPlore configuration there are 2 Index Agents for each docbase tunning on xPl-A and xPl-B, there are two queues created (trash\_ftindex\_00 and trash\_ftindex\_01): ftindex\_00 is running on vm-527b-5af5 and ftindex \_01 is running on vm-e62d-0840. You can eximine indexing stasus og each queue item by selecting the appropriate index on the Index Queue page below.

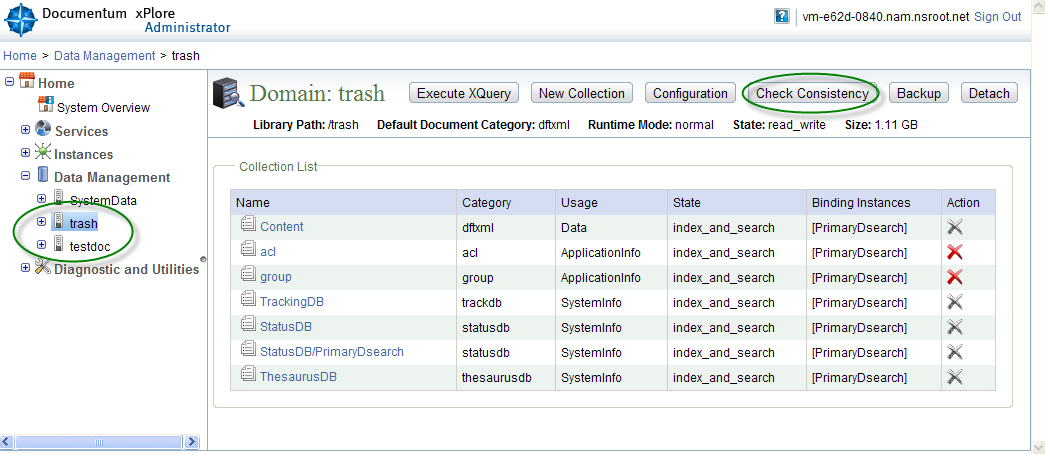
****

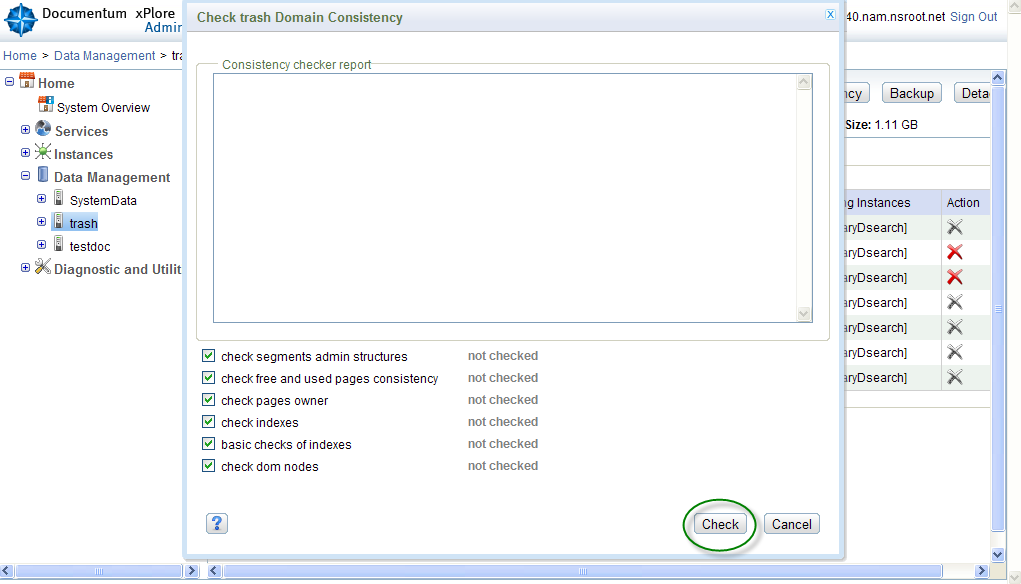
Next screenshot, shown a document, which failed indexing (Status: Indexing Failed). Specifically, index agent running on vm-527b-5af5 failed to index this document.

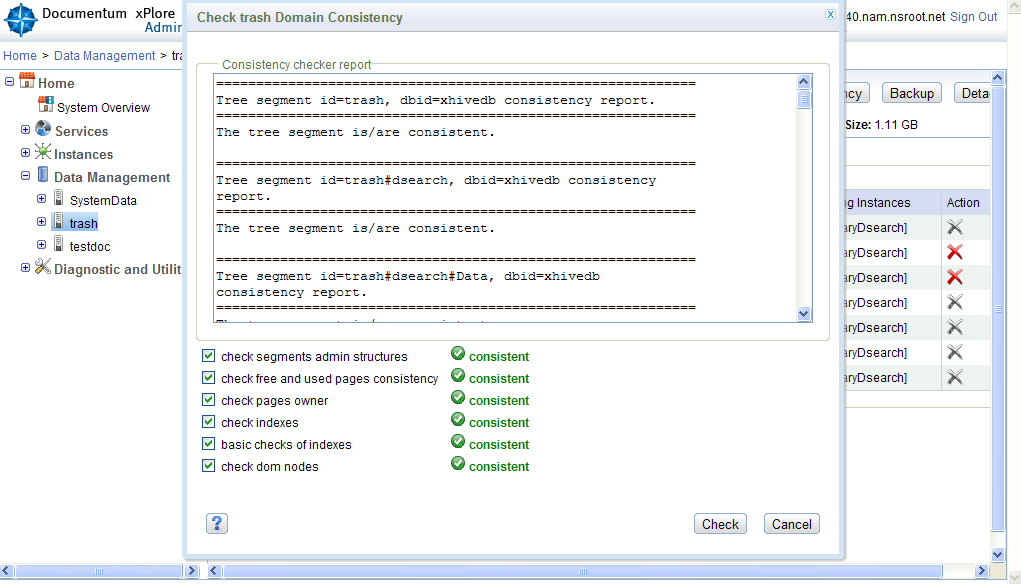
Resubmit this failed document to indexing again.

## Verifying integrity of indexes.

Login to dsearchadmin comsole (http://<xPlore-host>/dsearchadmin/) and for each domain (docbase), run ‘Check Consitency’.





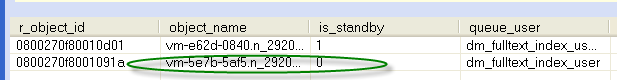


## Re-directing FT Searches to a standby xPlore host.

Only one xPlore host (Primary) is available for client searches (like webtop). If this host becomes un-available you need to reconfigure and restart all affected docbases on both CS-A and CS-B.

If one of HA xPlore Hosts goes down, query the docbase to find out which xPlore is NOT on standby (available for client quiries)

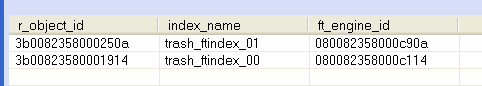
**select** ia.r\_object\_id,ia.object\_name,fi.is\_standby,ia.queue\_user **from** **dm\_ftindex\_agent\_config** ia,dm\_fulltext\_index fi **where** ia.index\_name=fi.index\_name



If vm-e62d-0840 (not on standby) is down, follow these steps to force client searches to be redirected to stand-by xPlore index collections.

1. Using IDQL run following:

**select** r\_object\_id,index\_name,ft\_engine\_id **from** **dm\_fulltext\_index**



1. Swap ft\_ingine\_id values for ftindex\_00 and ftindex\_01 above.

Using IAPI run (adjust r\_object\_ids):

execsql,c,update dm\_fulltext\_index\_s set i\_vstamp = i\_vstamp+1, ft\_engine\_id='080082358000c114' where r\_object\_id='3b0082358000250a'

execsql,c,update dm\_fulltext\_index\_s set i\_vstamp = i\_vstamp+1, ft\_engine\_id='080082358000c90a' where r\_object\_id='3b00823580001914'

## Missing Indexes. Identifying docbase content which has no corresponding indexes in xPlore.

Indexes count on matching domain/collections on both xPl-A and xPl-B should be the same. If not, you need to identify which objects in the docbase have no indexes in the xPlore and re-index them again.

On xPl-A host (active), change to /home/dmadmin/xPlore/setup/indexagent/tools folder.

There are ftintegrity\_for\_<docbase>.sh scripts for each domain which is configured on the host.

Edit this script (changes highlighted) , and specify the docbase user/password:

$JAVA\_HOME/bin/java -classpath "./lib/stateofindex.jar:$CONFIG\_DIR:$DOCUMENTUM/dfc/log4j.jar:./lib/httpcore-4.0.1.jar:./lib/httpclient-4.0.1.jar:./lib/commons-logging-1.1.1.jar:./lib/commons-codec-1.3.jar:./lib/commons-pool-1.6.jar:$DOCUMENTUM/dfc/dfc.jar:$DOCUMENTUM/dfc/aspectjrt.jar:./lib/dsearch-client.jar:./lib/dsearch-utils.jar:./lib/dsearch-install.jar:./lib/slf4j-api-1.6.4.jar:$DOCUMENTUM/dfc/activation.jar;$DOCUMENTUM/dfc/jaxb-api.jar:$DOCUMENTUM/dfc/jaxb-impl.jar:$DOCUMENTUM/dfc/jsr173\_api.jar" com.documentum.server.impl.method.fulltext.stateofindex.FTStateOfIndex trash doctester Welcome1 vm-5e7b-5af5.nam.nsroot.net 9300 $DSEARCH\_DOMAIN $OUTPUT\_FILEPATH $BATCH\_SIZE –useFilter T

**Note**: xPl-A host is the one identified by dm\_ftengine\_config:dsearch\_qrserver\_host attribute of the ft engine which is active (is\_standby=0).

Use following DQL/API to identify this host:

* 1. **select** e.r\_object\_id **from** **dm\_ftengine\_config** e, **dm\_fulltext\_index** fi **where** e.r\_object\_id=fi.ft\_engine\_id **and** fi.is\_standby=0
  2. **dump,c,<r\_object\_id-from above>**

param\_name [0]: dsearch\_qrserver\_protocol

[1]: dsearch\_qrygen\_mode

[2]: dsearch\_qrserver\_target

[3]: dsearch\_qrserver\_port

[4]: dsearch\_config\_port

[5]: dsearch\_qrserver\_host

[6]: dsearch\_domain

[7]: dsearch\_config\_host

[8]: query\_plugin\_mapping\_file

[9]: fast\_wildcard\_compatible

param\_value [0]: HTTP

[1]: both

[2]: /dsearch/IndexServerServlet

[3]: 9300

[4]: 9300

[5]: vm-5e7b-5af5.nam.nsroot.net

[6]: testdoc

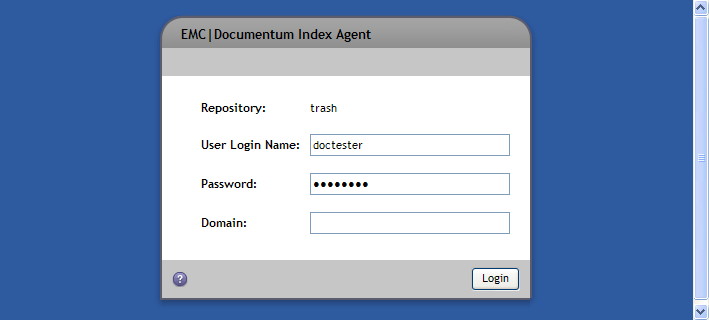
[7]: vm-5e7b-5af5.nam.nsroot.net

[8]: /opt/dmadmin/CS67SP1/fulltext/dsearch/dm\_AttributeMapping.xml

[9]: true

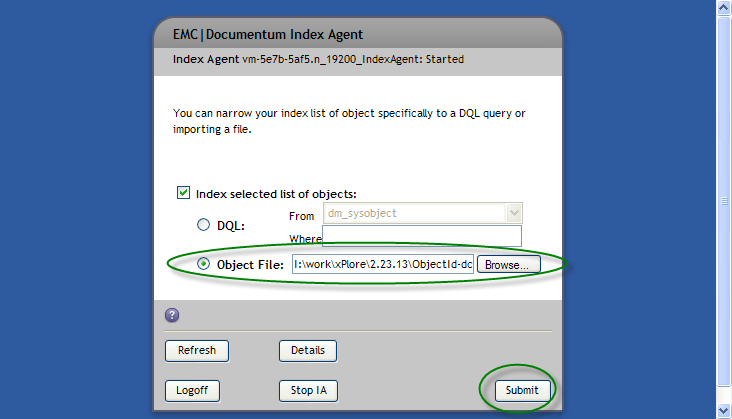
Use following sequence: this has to be run for all docbases for which domains exist on xPl-A

1. cd /home/dmadmin/xPlore/setup/indexagent/tools
2. ./ ftinegrity\_for\_<docbase>.sh
3. This will generate /home/dmadmin/xPlore/setup/indexagent/tools/ObjectId-dctmOnly.txt.
4. Transfer this file to your local PC
5. Login to IA console for docbase identified in Step 2: <http://xPl-A:19100/IndexAgent/login_dss.jsp>



If IA is running, stop it first

1. Select ‘Object File’ Radio Button, Browse to ObjectId -dctmOnly.txt on your PC which you saved in Step5 and ‘Submit’



For running ftinegrity\_for\_<docbase>.sh on xPl-B, follow following step. Repeat this for each docbase which has a domain on xPl-B.

1. Retrieve the secondary dm\_fulltext\_index object ID with this DQL statement:

select ft\_engine\_id from dm\_fulltext\_index where is\_standby=1

1. On both the primary and secondary Content Server, edit the server.ini file located in *documentum\_home*/dba/config/*servername*. Under [SERVER\_STARTUP], set the ftengine\_to\_use value to the secondary dm\_fulltext\_index object ID.
2. Restart both Content Servers.
3. Repeat steps 1 through 6 in preceding Section.

## File System Backup and Restore

Do daily (midnight) backups of both xPlore installations. Ther procedure tarball whole /home/dmadmin/xPlore folder after xPlore Instance Indexing Service is temporarily suspended.

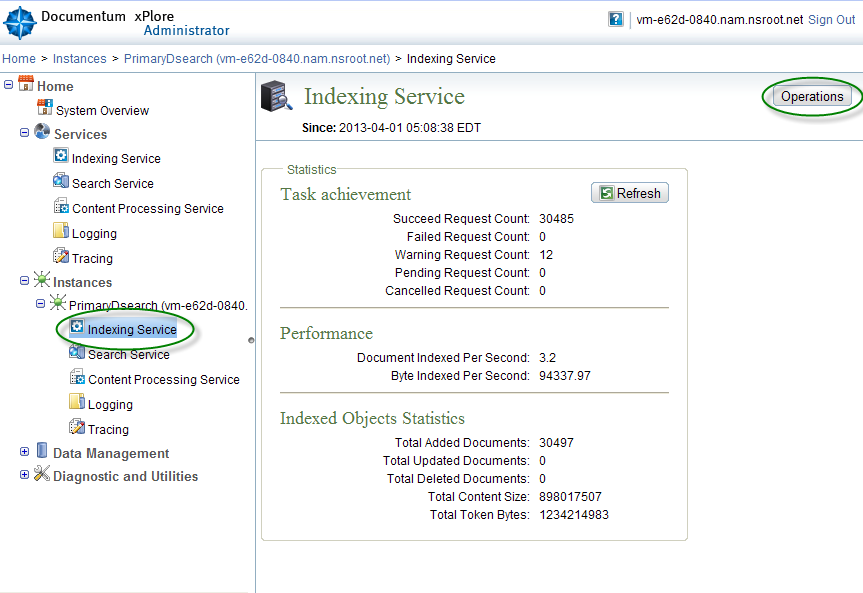
This insures that:

* Client searches are available throughout the procedure
* Any new content created in the docbase while Indexing Service is disabled, will be indexed automatically, after Indexing Service is enabled again. The will be no data loss.

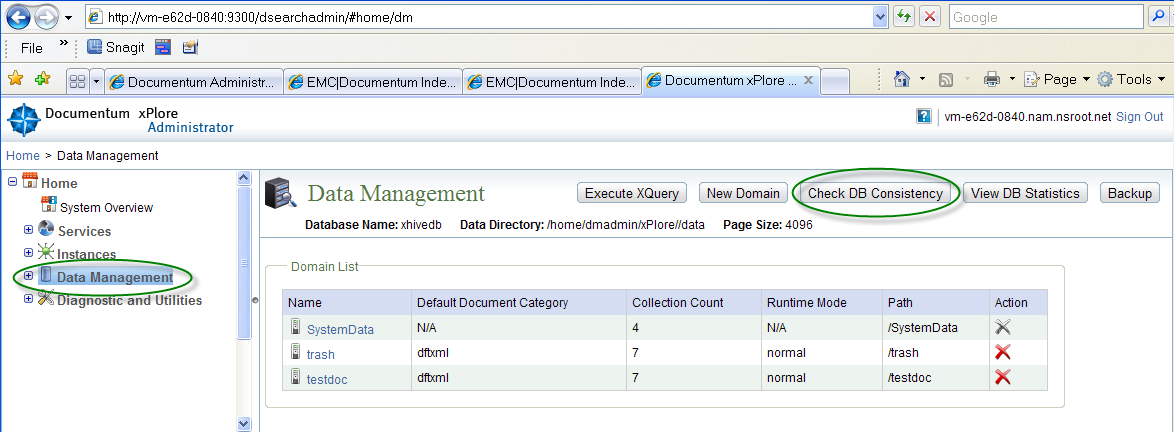
## Backup

## To perform FS xPlore backup, follow these steps:

1. *Suspend indexing service on xPlore Instance. Choose the instance and click Indexing* Service > Operations > Disable.

1. Run DB consistency on Data Management node of xPlore Admin console



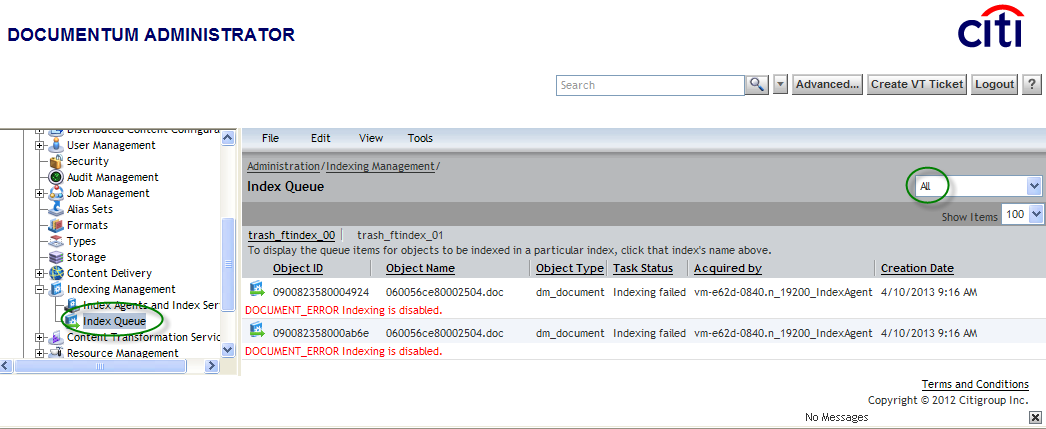
1. cd /home/dmadmin
2. tar -zcvf /nas/xPlore/<host>/xPlore<date>.tar.Z xPlore
3. After tar is done, enable xPlre Indexing Service, make sure all Index Agent are started, if appropeiate.
4. The whole procedure takes around 30 min.

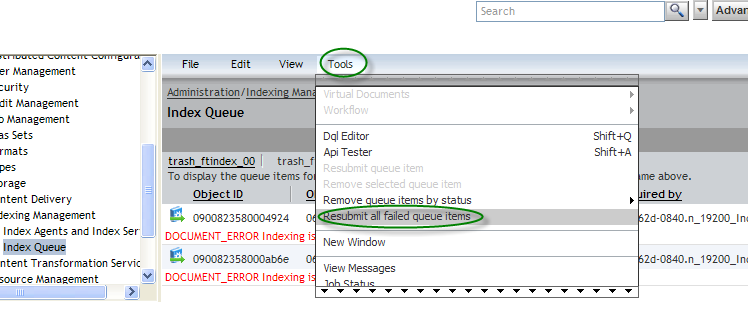
## Restore.

If xPlore installation gets somehow corrupted, you can restore entire /home/xPlore folder from the latest tarball.

After restoring xPlore installation from the tarball, start Primary Dsearch Instance and Run DB consistency Check on the Data Management node of the dsearchadmin to verify that system is OK.

1. On a failed xPlore host, backup and delete bad /home/dmadmin/xPlore folder
2. Untar latest xPlore tarball in /home/dmadmin folder
3. Start Dsearch and Index Agents on the rebuild xPlore host
4. Run DB consistency on Data Management node of xPlore Admin console
5. Start Index Agents in the Normal Mode (unless they were in Re0Indexing mode at time of xPlore failure. Go to each IA console (<http://host:n9200/IndexAgent/login_dss.jsp>) and Start Agent in normal mode.
6. Resubmit failed indexing jobs (because of suspended Indexing Service). Go To DA and select Index Queue. If you see any docs with ‘Index Failed’ Status, select Tools and resubmit them.





## Backing up and restoring xPlore Federation.

This procedure will prepare a dump of all configured domains on xPlore host in a binary format. The dump location is specified in following element of /home/dmadmin/xPlore/config/indexserverconfig.xml

<admin-config>

<backup-location path="/nas2/xPlore/backup/vm-5e7b-5af5"/>

</admin-config>

## Backup.

## Restore.

This dump can be later restored to the xPlore host from which it was originally made.

Restoring saved Federation dump requires xPlore to be completely shutdown.

To restore saved Fedration dump, follow this steps

1. Stop xPlore Primary instance and IndexAgents on the host

2. Backup /home/dmadmin/xPlore : ]$ tar -cvzf xPlore.tar.gz xPlore

3. Delete existing data files:

All files under /home/dmadmin/xPlore/data

All files under /home/dmadmin/xPlore/config

4. Restore Federation backup

cd /home/dmadmin/xPlore/dsearch/admin

ksh -x ./xplore.sh "restoreFederation =’/nas2/xPlore/backup/vm-5e7b-5af5/federation/2013-04-01-07-57-52' "

5. Restart xPlore Primary instance and start all IndexAgents is Normal Mode.

## Restoring data after an instance fails

In HA setup, you can restore data from one instance to another when one of the instances fails.

The following steps use two example instances: InstanceA and InstanceB. xPlore is set up on these two instances with mirrored information: same folder structures, same username/password, same instance name, same ports, etc. The example assumes that data corruption occurs on xPl-B. If it was xPl-A, swap xPl-A with xPl-B

1. Stop xPlore and IndexAgent on xPl-A. Record the time the index agent is shut down.
2. Stop xPlore and IndexAgent on xPl-B.
3. Backup xplore\_home/config/indexserverconfig.xml and xplore\_home/config/XhiveDatabase.bootstrap on xPl-B. You will use this backup to recover configuration information after you restore data.
4. Copy the directories xplore\_home/data and xplore\_home/config from xPl-A to xPl-B. Do not change file permission.
5. Start xPlore and the index agents in normal mode (not migration mode) on xPl-A.

Update xplore\_home/config/indexserverconfig.xml on xPl-B with settings in the original backup copy. Change the attributes “url” and “hostname” of the XML element index-server-configuration.node to match those values in the backup file. Bump up xml revision, like like:

<index-server-configuration revision="1.40"

To:

<index-server-configuration revision="1.41"

1. Update xplore\_home/config/XhiveDatabase.bootstrap on xPl-B to match the original file. Change attribute “host” of the element /server/node to match the value in the backup file.
2. If xPl-B failed some time ago, you must clear many duplicate indexing tasks that are queued as dmi\_queue\_items. Run DQL to truncate the queued items for IndexAgent on xPlore-B: In the following example, IA\_user is the user for IndexAgentB, *date*\_value is a time just before the index agent shutdown time recorded in step1.

?,c,delete dmi\_queue\_item objects where name='IA\_user' and date\_sent<=date('*date*\_value')

For example, if xPl-B was stopped due to failure on 3.22.2013:

?,c,delete dmi\_queue\_item objects where name='dm\_fulltext\_index\_user01'

and date\_sent<=date('5/22/2013 21:24:45')

1. Start xPlore and the index agent in normal mode on xPl-B.

## Testing search

Before querying the index, install a Documentum index agent for xPlore and index some content, or upload content using xPlore administrator.

1. In xPlore Administrator, select Diagnostic and Utilities > Test Search.
2. Choose one:
   * Keyword: Specify the text for which you want to search.
   * XQuery: Enter a query using XQuery syntax.
3. Select a domain, collection, language, and maximum number of results and then click Search.
4. Verify that the correct results are displayed.
   * If you use a Webtop client to test search, click Control-Edit to see the XQuery that was used in the search.
   * To check whether a search is executed against FAST or xPlore, use the ENABLE(FTDQL\_DATA) hint. For example, issue this iAPI command (one a single line):
   * ?,c,SELECT r\_object\_id FROM dm\_sysobject

SEARCH DOCUMENT CONTAINS 'dmadin' ENABLE(FTDQL\_DATA)

1. If the Content Server executes against xPlore, the result contains an XQuery snippet. For example:

...Native Query:

[((. ftcontains( (('dmadmin') with stemming) ) ))

and ( ( (dmftinternal/i\_all\_types = '0300271080000105') )

and ( (dmftversions/iscurrent = 'true') ) )]

## Removing entries from the index

You can remove certain object types, or objects that meet other criteria such as dates, from the index. You can execute a DQL query to get object IDs of the documents that you wish to delete from the index. Save the list of object IDs in a text file.

1. Navigate to xplore\_home/dsearch/xhive/admin.
2. Open deletedocs.properties in a text editor.
3. Make sure that the host and port values correspond to your environment.
4. Set the value of dss\_domain to the xPlore domain from which you wish to delete indexed documents.
5. Change the value of the key file\_contains\_id\_to\_delete to the path to your object IDs. Alternatively, you can list the object IDs, separated by commas, as the value of the key ids\_to\_delete.
6. run /home/xPlrore/dsearch/xhive/admin/deleteDocs.sh.

The deletedocs utility records activity in a log in

/home/xPlrore//dsearch/xhive/admin/logs/deleteDocs.log

## Document processing (CPS) reports

Run the Document processing error summary report to find the count for each type of problem. The error count for each type is listed in descending order. The following types of processing errors are reported: request and fetch timeout, invalid path, fetching errors, password protection or encryption, file damage, unsupported format, language and parts of speech detection, or document size.

View detailed reports for each type of processing error. For example, the Document processing error detail report for Error code 770 (File corrupt) displays object ID, domain, date, time, format, and error text. You can then locate the document in xPlore administrator by navigating to the domain and filtering the default collection for the object ID. Using the object ID, you can view the metadata in Content Server to determine the document owner or other relevant properties.

Run the report Content too large to index to see how many documents are being rejected for size. If your indexing throughput is acceptable, you can increase the size of documents being indexed. Run the report User activity to see ingestion activity and error messages for ingestion by a specific user and time period.

# COB SETUP

This procedure assumes that COB content server is pointing to Prod Oracle SID (server.ini: database\_conn) and Prod NAS filestore. Docbroker ports are identical in both COB and Prod.

COB file system setup is identical to Prod setup, except that they are in different DC. COB Content Servers share the same Oracle SID and same content filestores on NAS.

In general, when failing over from Prod to COB you have to

* Modify (IAPI/DQL) some docbase object
* Shutdown Prod docbase and start COB docbase.
* Clone Prod xPl-A and xPl-B File Systems to COB xPl-A and xPl-B hosts
* Start COB docbase

Keep accurate record of changes you are performing during COB fail-over.

## Failing-over from Prod to COB

1. Login into Prod DA pointing to Prod CS. If Prod CS are down, log inot COB CS.
2. Dump out all the objects you modify, before you update them. You will need to restore these objects, when switching back to Production. Dump following objects and save these dumps:

* **dm** \_**ftengine\_config (2)**
* **dm\_server config (4)**

1. Modify Full text related objects in the docbase. This has to be done for each docbase configured for FT indexing in Prod xPlore.
   1. Modify **dm** \_**ftengine\_config:**

There are two of these in each docbase, each pointing to Prod xPl-A or Prod xPl-B respectively. We need to re-point them tp COM xPlore hosts.

Change dsearch\_qrserver\_host and dsearch\_config\_host parameters of each **dm**\_**ftengine\_config** to point to xPlore host in COB, instead of Prod.

* IDQL: **select** r\_object\_id **from** **dm\_ftengine\_config**
* For one of r\_object\_id from above, run:

set,c,<r\_object\_id#1>, param\_value[5],<FQN of COB xPl-A>

set,c,<r\_object\_id#1>, param\_value[7],<FQN of COB xPl-A>

save,c, <r\_object\_id#1>

* For the second r\_object\_id, run:

set,c,<r\_object\_id#2>, param\_value[5],<FQN of COB xPl-B>

set,c,<r\_object\_id#2>, param\_value[7],<FQN of COB xPl-B>

save,c, <r\_object\_id#2>

Note: When you will fail-over back to Prodenvoronment, you will have to restore each docbase object you modified for COB back to their original values (dumps you saved before the original failover).

* 1. Modify **dm\_server\_config**

There are 4 of these objects, 2 for each CS host. Each of these points (app\_server\_uri[2,3]] to xPlore hosts in Prod. We will re-point the to xPlore hosts in COB.

For example, if current Prod **dm\_server\_config** dump has:

app\_server\_uri

**[2]: http://vm-5e7b-5af5.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent**

**[3]:** [**http://vm-e62d-0840.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent**](http://vm-e62d-0840.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent)

Where vm-5e7b-5af5 and vm-e62d-0840are Prod xPlore hosts

You have to change them to:

app\_server\_uri

**[2]: http://vm-0000-0001.**nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent

**[3]:** [http://**vm-0000-0002**.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent](http://vm-0000-0002.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent)

Where **vm-0000-0001** and vm-0000-0002 are COB xPlore hosts.

Follow these steps:

* IDQL: **select** r\_object\_id **from dm\_server\_config**
* For each of 4 r\_object\_id from above, run

set,c,<r\_object\_id>, app\_server\_uri [2], **‘http://vm-0000-0001.**nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent’

set,c,<r\_object\_id>, app\_server\_uri [3, **‘http://vm-0000-0002.**nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent’

save,c, <r\_object\_id>

* 1. Shutdown COB docbase on Both COB CS host.

1. Untar recent backup of /home/dmadmin/xPlore folder from Prod xPlore hosts to /home/dmadmin/ folder on COB hosts. Backup from Prod xPl-A goes to COB xPl-B and so on
2. On each of xPlore hosts, modify some configuration files to reflect hostname change.

Cd /home/dmadmin/xPlore-1.3-SilentInstaller/script

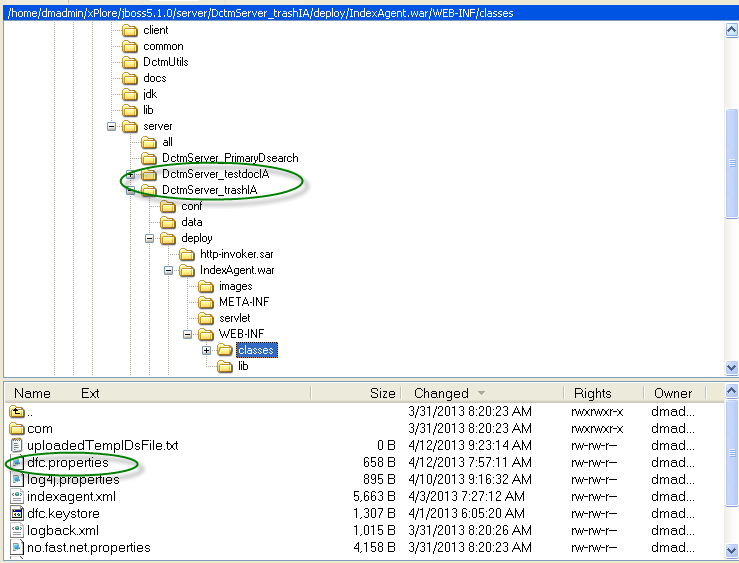
./ updateHostAfterReimaging.sh <hosname of COB xPlore host>.

Specify the new hostname without full domain name.

For example,

./ updateHostAfterReimaging.sh vm-0000-0001;

1. Modify IndexAgent’s dfc.properties to point to COB Content Server brokers. Each docbase has associated with it an Index Agent running on each xPlore host. Thus, there are 2 IndexAgents configured, you have to modify 2 dfc.property files on each of COB xPlore hosts.



1. Start COB docbases and run following DQL to verify that FT indexes are configured correctly:

**select** ia.r\_object\_id,ia.index\_name,ia.object\_name,fi.is\_standby,ia.queue\_user **from** **dm\_ftindex\_agent\_config** ia,dm\_fulltext\_index fi **where** ia.index\_name=fi.index\_name

1. Start Primary DSearch Admin consoles on both xPlore COB hosts.
2. Login to both xPlore Admin consoles. Under Data Management node, you should see all domains and collections as you saw the in Prod environment
3. Run DB Consistency Check on Data Management node.
4. On both COB xPlore hosts, start Index agents
5. Visit all Index Agent’s consoles, and start them in a normal mode. For, example, if there are 2 docbases configured, there are 4 IA consoles (2 on each xPlore host) to visit and start in normal mode.
6. Go to DA, select Index Management|Index Agents and Index Servers and make sure IA are running.
7. Import some docs and verify that that they are indexed on both COB xPlore hosts. Go to xPlore Admin console>>Data Management>>Domain>> Default Collection and goto the last page of the collection. You should see documents which you just imported.

## Failing over from COB to Prod.

1. With COB Content Servers still i[, restore docbase objects you have modified when failing over to COB, to their original state
2. Shutdown COB Content servers and xPlore hosts
3. Startup Prod docbases
4. Start Prod docbases and run following DQL to verify that FT indexes are configured correctly:

**select** ia.r\_object\_id,ia.index\_name,ia.object\_name,fi.is\_standby,ia.queue\_user **from** **dm\_ftindex\_agent\_config** ia,dm\_fulltext\_index fi **where** ia.index\_name=fi.index\_name

1. Start Primary DSearch Admin consoles on both xPlore Prod hosts.
2. Login to both xPlore Admin consoles. Under Data Management node, you should see all domains and collections.
3. Run DB Consistency Check on Data Management node.
4. On both Prod xPlore hosts, start Index agents
5. Visit all Index Agent’s consoles, and start them in a normal mode. For, example, if there are 2 docbases configured, there are 4 IA consoles (2 on each xPlore host) to visit and start in normal mode.
6. Go to DA (pointing to Prd), select Index ‘Management Index Agents and Index Servers’ and make sure IA are running.
7. Import some docs and verify that that they are indexed on both COB xPlore hosts. Go to xPlore Admin console>>Data Management>>Domain>> Default Collection and go to the last page of the collection. You should see documents which you just imported.

# APPEnDIX A. xplore installation example

This appendix describes engineering setup of xPlore in HA configuration in RHEL 62

* There are 2 Content Server hosts for each docbase:

**CS-A**: Primary CS host Connection broker: delabt04:1489

**CS-B**: Secondary CS host Connection broker: delabt30:1489

* There are 2 xPlore hosts with PrimaryDSearch Instance on each

**xPl-A**: Primary: vm-5e7b-5af5**. Participates in both indexing and searches**.

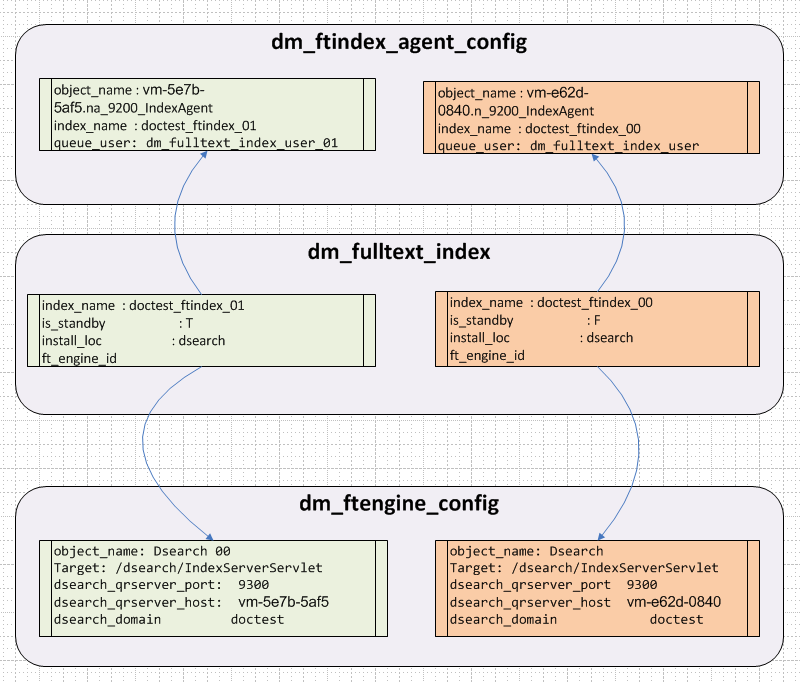
* 2 Index Agents: testdoc and trash
* Two domains: testdoc and trash. All ft indexes for docbase content are stored in Deefault collection.

**xPl-B**: Secondary (stand-by): vm-e62d-0840. Participates in indexing, like Primary xPl-A, but does not support searches. All client searches (webtop) are going against xPl-A index collection. If xPl-A is down, clients cannot execute Full Text searches. There is no automatic fail-over support for searches.

* 2 Index Agents: testdoc and trash
* Two domains: testdoc and trash. All FT indexes for docbase content are stored in Default collection.



**FT-related docbase objects**. Every time a new Index Agent is configured on xPlore host, 3 Full Text related objects are created in the Content Server host docbase:



Since we are configuring IA (for a specified domin/docbase) on both Primary and Secondary (xPl-A and xPl-B), 2 sets of above 3 objects are created in the docbase. Since we have 2 Content Server hosts and 2 xPlore Hosts in HA, there are two sets of these 3 objects: one (on left) is for doctest IndexAgent running on Primary xPl-A, the other (on the right) is for the Secondary xPl-B.

The result of this is that both xPl-A and xPl-B index same content independently- index collection sizes should be identical on both xplore hosts. If one of xPlore instances fails, the other xPlore instance will go on indexing to its own collection. Eventually, you will have to catch up and re-index the missed content indexing. The procedure for this is described [here](#_Identifying_docbase_content)

But for searches the situation is different: Only one of two dm\_fullttext\_index can do searches - this is specified in dm\_fulltext\_index:is\_standby attribute. In the Figure above, doctest\_ftindex\_01 (on the left) is on standby, which means that all client searched (from webtop) are routed to xPl-B (vm-e62d-0840) collections (/testdoc/Content). In this case, docbase configuration has to be changed to fail-over client searched from from xPL-A to xPL-B. This procedure is described [here](#_Re-directing_FT_Searches).

Additionally, Index Agent name and servlet URI as appended to all dm\_server\_config objects for each docbase which is being indexed. Notice that you can't tell from looking at these values, which domain (docbase) it belongs to. In our example, we assigned port 19200 to 'testdoc' IndexAgent when [configuring](#_Configure_Index_Agents.) IndexAgent. When we added IndexAgent for 'trash' domain, we assigned it port 29000. Keep a record of your actual assignment. Maintain identical IA port assignments across dev/uat/prod/cob.

Notice, that there are two Index Agents for every docbase

One is running on xPl-A and connects to the docbase though docbroker on CS-A.

The second IA is running on xPl-B and connects to the docbase though docbroker on CS-B

|  |  |  |  |
| --- | --- | --- | --- |
| Domain/Docbase | IA Name | IA Port | IA URI |
| trash on delabt04:1489 broker | vm-5e7b-5af5.n\_19200\_IndexAgent | 19200 | http://vm-5e7b-5af5.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent |
| trash on delabt30:1489 | vm-e62d-0840.n\_19200\_IndexAgent | 19200 | http://vm-e62d-0840.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent |
| delabt04:testdoc | vm-5e7b-5af5.n\_29200\_IndexAgent | 29200 | http://vm-5e7b-5af5.nam.nsroot.net:29200/IndexAgent/servlet/IndexAgent |
| delabt30:testdoc | vm-e62d-0840.n\_29200\_IndexAgent | 29200 | http://vm-e62d-0840.nam.nsroot.net:29200/IndexAgent/servlet/IndexAgent |
|  |  |  |  |
|  |  |  |  |

dm\_server\_config app\_server\_name[2] : vm-5501-020F.n\_19200\_IndexAgent

dm\_server\_config app\_server\_uri [2]: http://vm-5501-020f.nam.nsroot.net:19200/IndexAgent/servlet/IndexAgent

# APPEnDIX B. xplore Start/stop scripts

## Dsearch Admin console. Primary instance

cd /home/dmadmin/xPlore/jboss5.1.0/server;nohup ./ startPrimaryDsearch.sh &

cd /home/dmadmin/xPlore/jboss5.1.0/server;nohup ./ stopPrimaryDsearch.sh &

## Index Agents

cd /home/dmadmin/xPlore/jboss5.1.0/server;nohup ./start<docbase>IA.sh &

cd /home/dmadmin/xPlore/jboss5.1.0/server;nohup ./stop<docbase>IA.sh &

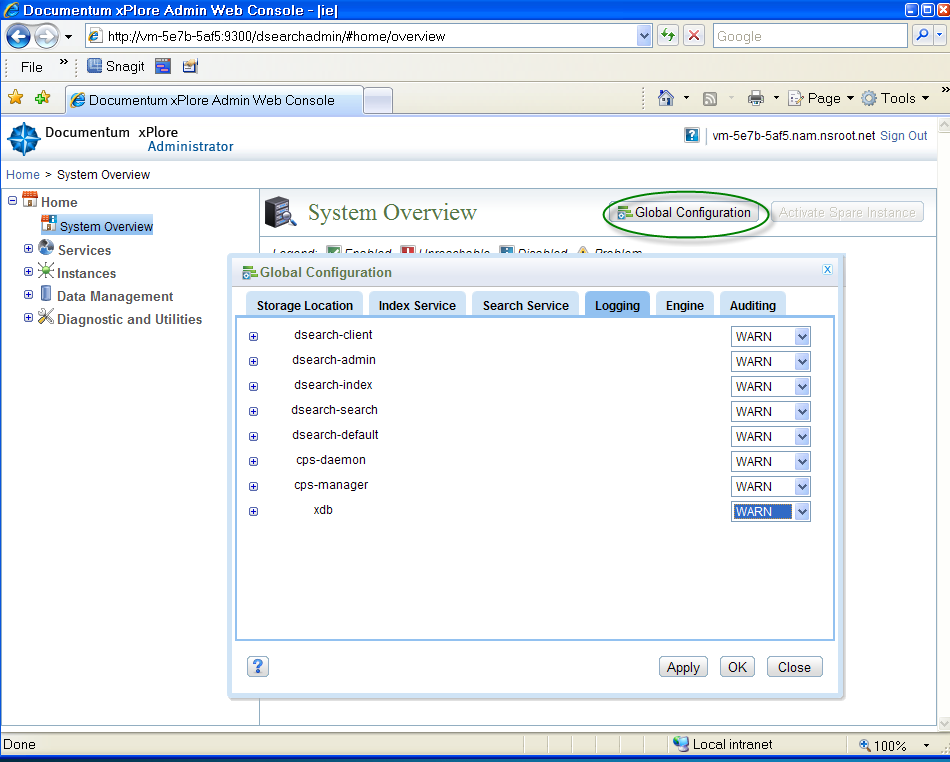
## Watchdog

cd /home/dmadmin/xPlore/watchdog/;nohup ./startWatchdog.sh &

cd /home/dmadmin/xPlore/watchdog/;nohup ./stopWatchdog.sh &

# APPEnDIX c. xplore logs

Logging levels are set in *Global Configuration* Panel of xPlore Admin Portal (http://host:9300/dsearchadmin)



Location of various log files is as follows:

* **Search**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/logs/ dsearch.log
* **CPS**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/logs/ cps\_daemon.log
* **xDB**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/logs/ xdb.log
* **Index**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_<docbase>IA/logs/ <docbase>IA.log
* **xPlore Instance**: /home/dmadmin/xPlore/jboss5.1.0/server/DctmServer\_PrimaryDsearch/log/server.log

# APPEnDIX D. External CPS.

By default, every xPlore instance has a local CPS service. Each CPS service receives processing requests on a round-robin basis. For a high-volume environment with multiple xPlore instances, you can configure a dedicated CPS for each instance. You must have low network latency. This dedicated CPS reduces network overhead. See [Configuring CPS dedicated to indexing or search](http://vm-5e7b-5af5:9300/dsearchinfocenter/topic/com.emc.dsearch.admin.help/topics/en-us/inst_t_CPSPerInstance.html).

To improve indexing or search performance, you can install CPS on a separate host. The installer adds a JBoss instance, CPS ear file, and CPS native daemon on the remote host.

Follow these steps to configure external CPS:

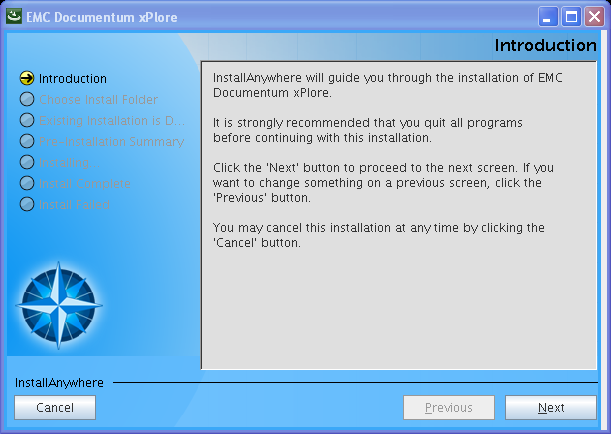
For this step you will need xServer. Refer to Section 2.2.1 for details

1. Logon in as dmadmin to xCPS host.
2. Configure swapspace
   1. dd if=/dev/zero of=/swapfile bs=1M count=4245
   2. /sbin/mkswap /swapfile
   3. /sbin/swapon /swapfile
3. Add following lines to /etc/fstab (substitute your actual filer for delabtnas01:/vol/vol\_01). This filer volume has to be the same you setup on xPl-A and xPl-B hosts.

delabtnas01:/vol/vol\_01 /nas nfs rw,hard,intr 0 0

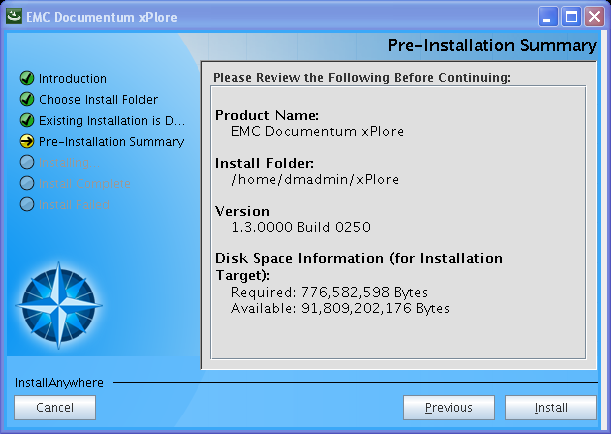
/swapfile swap swap defaults 0 0

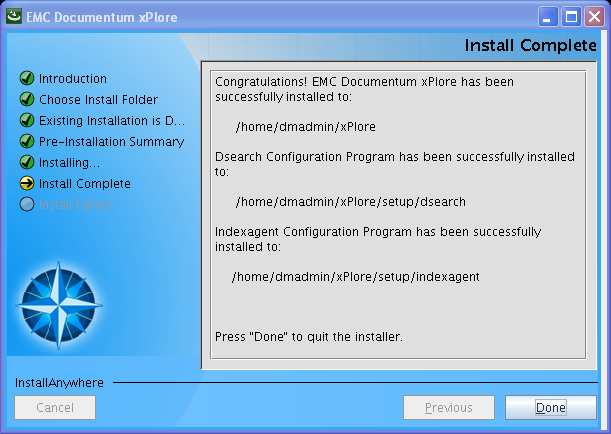
1. Mount NAS filer: mount -va
2. Setup DISPLAY variable: export DISPLAY=xx.xx.xx.xx:0 and start xServer (XWIN32.EXE)
3. cd /home/dmadmin/xPlore-1.3-SilentInstaller/xPlore-1.3-setup
4. run ./setup.bin



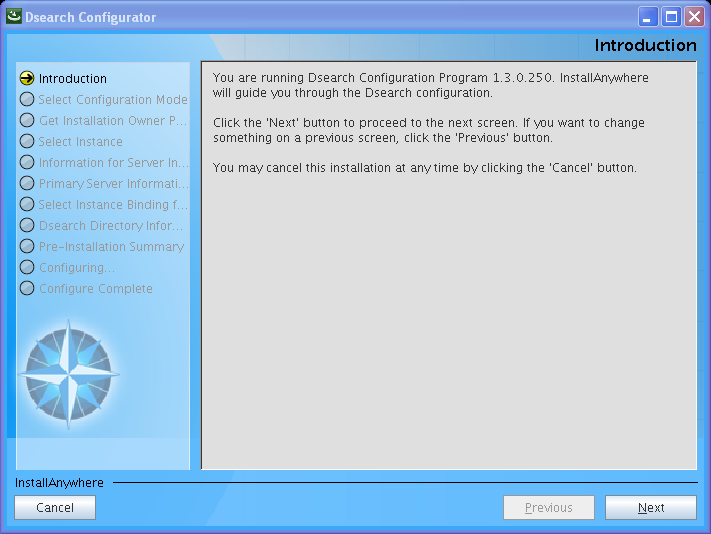


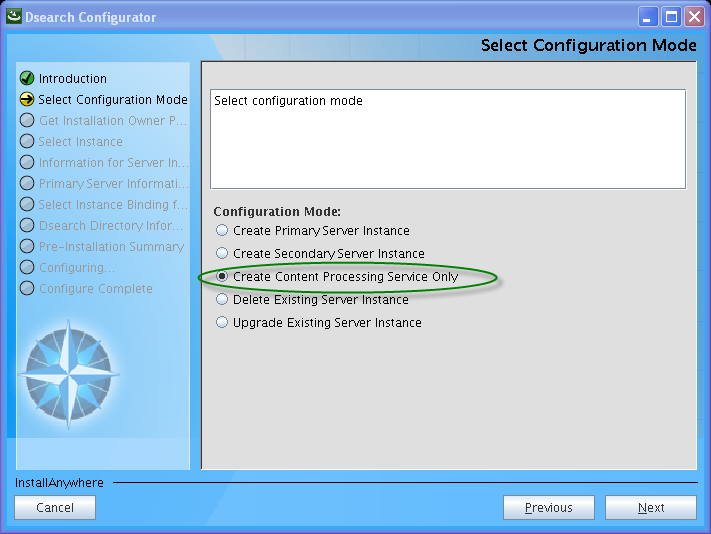


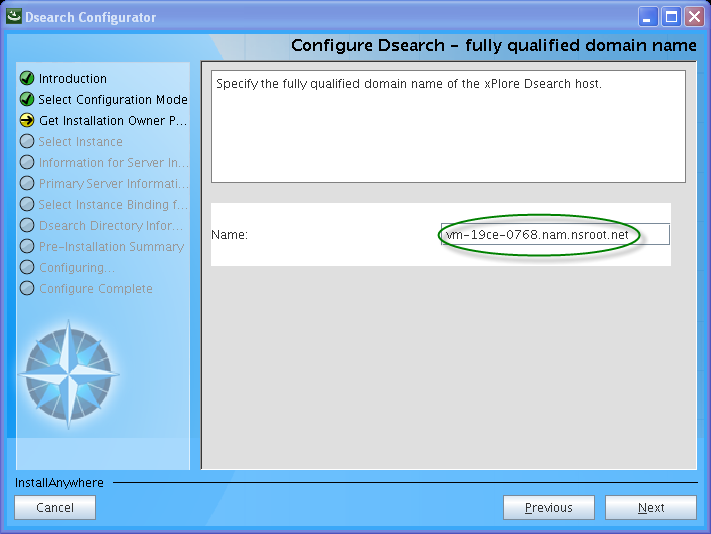




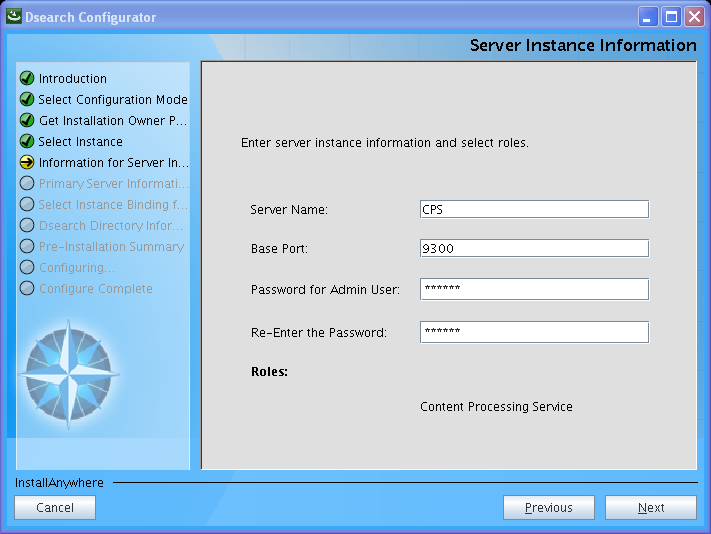
1. cd /home/dmadmin/xPlore/setup/dsearch;./configDsearch.sh



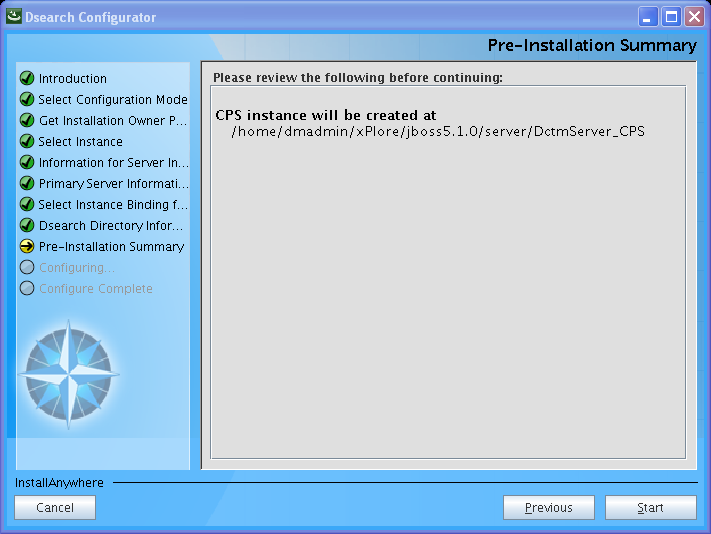


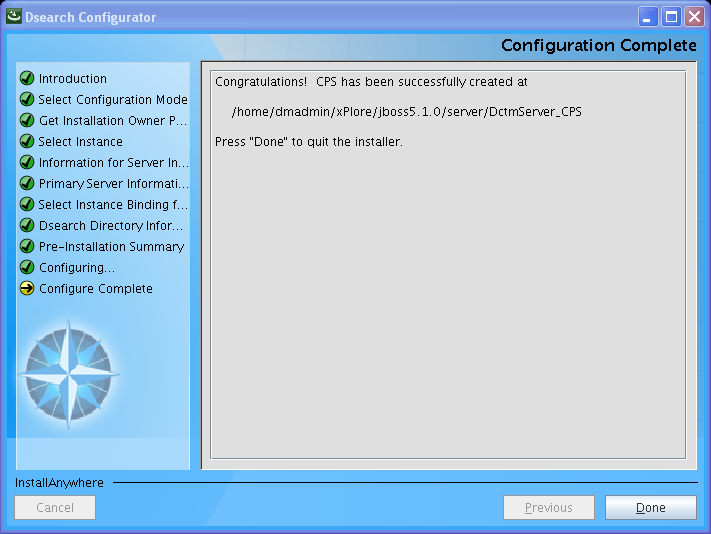


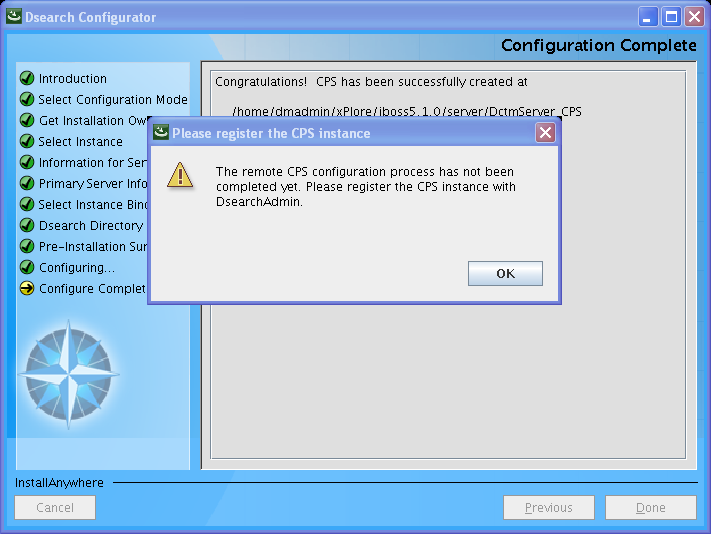
The name of the actual host will show up



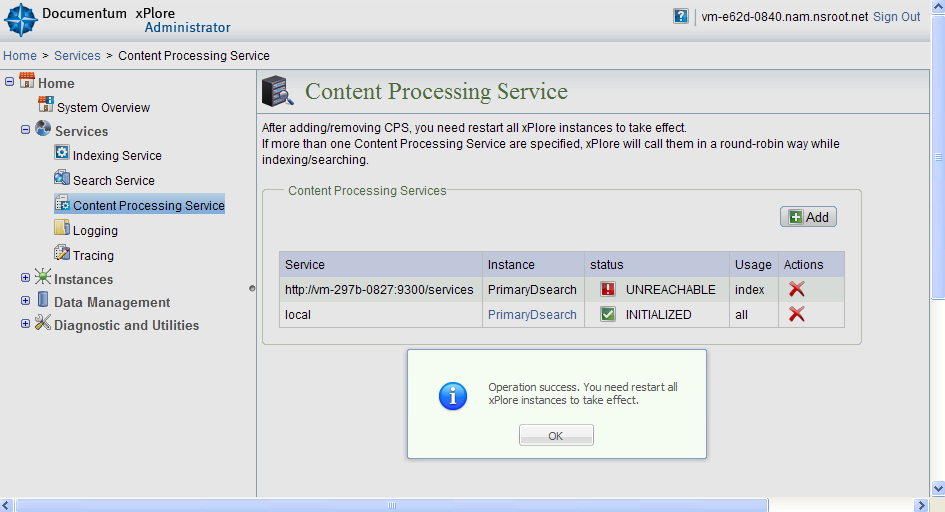
Enter qw12!@ for Password







Click OK. You will register CPS instance in next step



1. vi /home/dmadmin/xPlore/dsearch/cps/cps\_daemon/CPS\_configuration.xml:

Change

<export\_file\_path>[file:///home/dmadmin/xPlore/dsearch/cps/cps\_daemon/export</export\_file\_path](file:///\\CPNARUTAP001\home\dmadmin\xPlore\dsearch\cps\cps_daemon\export%3c\export_file_path)>

To

<export\_file\_path>file:/// nas/xPlore/< xCPS >/cps/cps/export</export\_file\_path>

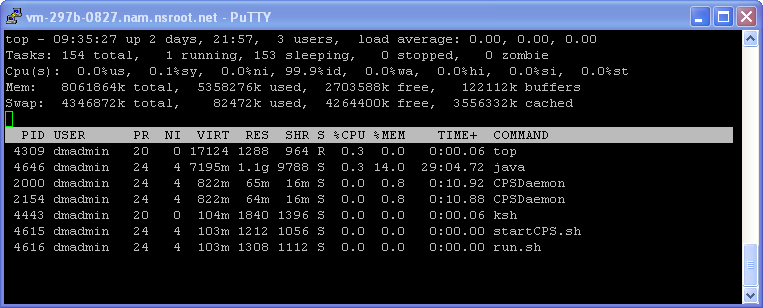
Substitute actual hostname for < xCPS >

1. Start CPS:

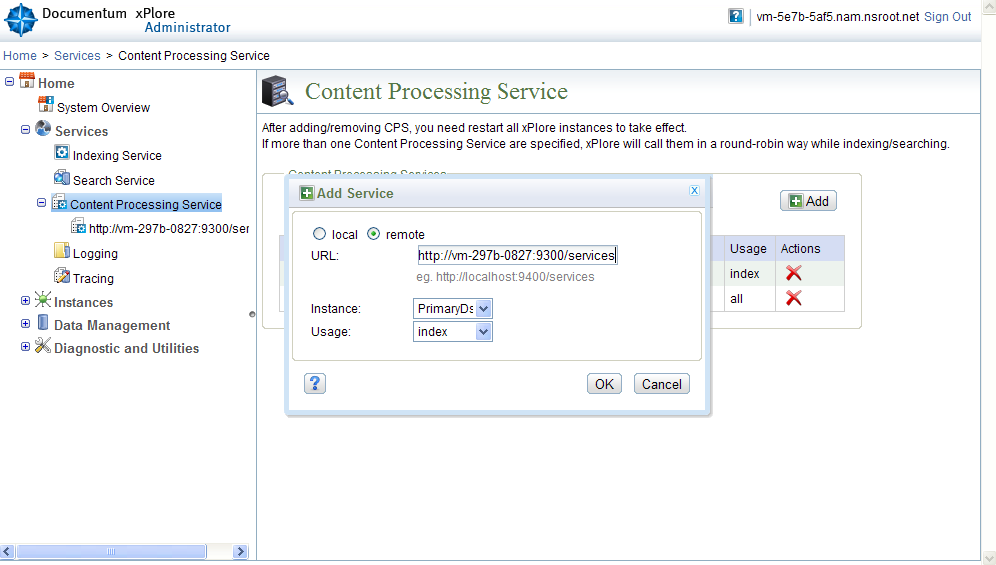
Cd cd /home/dmadmin/xPlore/jboss5.1.0/server

Nohup ./startCPS.sh &

1. Verify: top –U dmadmin



1. Register Remote CPS with Primary DSearch instance on both xPl-A and xPl-B:



Substitute actual remote CPS hostname in the URL

1. Restart remote CPS and all instances and IA on xPl-A and xPl-B
2. Verify that remote CPS was registered on DSearch Primary Instance on both xPl-A and xPl-B:

