EMC²
where information lives®

Understanding In-Depth Documentum Security

Narsingarao Miriyala

EMC Proven Professional Knowledge Sharing 2010

EMC² PROVEN PROFESSIONAL

Narsingarao Miriyala
EMC Corporation
Miriyala_narsingarao@emc.com

Contents

Table of Figures

Background

I have designed Enterprise Content Management Systems (ECMS) and led content strategy for corporations of all sizes. Business and IT stake holders of ECM projects often ask system architects to achieve security accreditation from the Office of Information Security (OIS) in addition to certifying the system design and solution by the Corporate Architecture Review Board. Fulfilling these two requests is not trivial. Currently, there are tools and documentation available within our organization to present a well defined and detailed system design to the Architecture Review Board.

But when it comes to presenting the system design for OIS accreditation, it is daunting due to the lack of proper procedures, tools and documentation. Security accreditation of a system design would not be so challenging if we could list all of the security related information in a prescribed document. This is my effort to develop such a document for OIS accreditation for all Documentum® applications and solutions.

Introduction

A Documentum repository can contain a corporation's most sensitive assets such as Financial, HR, Product roadmaps, Product patents or Service offerings data / documentation. Every company's OIS team guards their repository assets from unauthorized access by users or system administrators. Corporations want to protect their vital data during transmission as well as when in storage. This article will not cover Documentum basic security like ACLs, groups, roles etc. as all of these topics are extensively covered in Content server documentation. I am going to cover the topics that are difficult to find in the documentation. This article is written from a Documentum 6.5 release perspective.

Audience

This article is written for customers, partners and consultants validating Documentum infrastructure for Security certification. It can be used by technical team as reference material to build the Documentum infrastructure to meet business requirements. It is assumed that the audience is familiar with the Documentum repository.

Terminology

ACL – Access Control List, ACL's are applied on object level

ACS – Accelerated Content Services, ACS server is used to read/write content files from user browser to storage. ACS is part of Content Server embedded JBoss application server.

DFC – Documentum Foundation Client, All WDK or Custom clients use DFC to connect to Repository.

OIS – Office of Information Security

TCS – Trusted Content Services

UCF – Unified Client Framework, UCF is deployed on Client Machine as Java Applet. UCF communicates with Application Server and ACS server for all Content related operations like import/export/check-in/check-out.

WDK – Web Development Kit, it is a framework. WebTop, DAMTop, Documentum Administrator (DA) are developed using WDK.

WebTop – Documentum Web Client developed using WDK framework

DA – Documentum Administrator, DA is used by Documentum admin person to administer the repository.

Documentum Repository Security

Documentum Repository Security is categorized as the following”

- Identity Management
- Authentication
- Authorization
- Trusted Content Services (TCS)
- Auditing

Identity Management

Documentum repository can have the following set of users:

- In-Line users
- LDAP users
- Local OS users
- Windows Domain users

In-Line Users

Documentum Administrators create Inline users, encrypt their passwords, and store them in the repository. This is an easy way to create users in the repository, although it is less secure than other options.

LDAP Users

You can configure the Documentum Repository to integrate with Corporate Directory Services to synchronize users and groups. Documentum supports all the major LDAP directory services. The Repository supports synchronizing users and groups from multiple directory servers. The system can be configured to secure the transmission (SSL) between the repository and LDAP server. The system allows synchronizing multiple user and group object attributes from the LDAP server and stored in Repository as dm_user/dm_group or a subtype of dm_user/dm_group objects. User passwords are not retrieved from the LDAP server. The Repository will connect to LDAP server for user authentication.

The Synchronization job also updates/disables users in the repository if a user is disabled in the LDAP server or if user attributes are updated (like last name) in LDAP.

Local OS Users

Documentum can be configured to authenticate users on the host OS; administrators create user objects in the repository corresponding to local users on underline OS for the Content Server. The Repository will authenticate the user with the underlying OS.

Windows Domain Users

You can configure a Repository running on Windows OS to authenticate users with the Windows domain. The Administrator has to create domain users manually in the repository before users can login to Documentum.

Authentication

As I explained in the previous section, Documentum will not store the user password in the repository (except for in-line users). The user must provide userid, password, and domain name (optional). The repository takes the information provided by the user and authenticates it with the corresponding authentication mechanism based on the user info. If the user is from an LDAP server, the repository will authenticate the user with the LDAP server. If the user is created as a Windows domain user, the repository will authenticate the user with the Windows domain, similarly for OS users.

Documentum also has built in additional mechanisms for authorized access to the system such as:

- Session Timeout – System will timeout a user session and release all the resources held by user if the user is inactive for a period of time. The Inactive period can be configurable.
- The System can be configured to lock the user if he/she enters the wrong password for a pre-defined number of times. This functionality also addresses the ***Brute Force Attacks*** issues.
- The System can be configured to record audit events when a user logs into the system. The System captures the date and the time when the user logged into the system. It can also capture unsuccessful logins as part of audit.

Authorization

The Documentum Repository allows users to access assets at an object level using the Access Control List (ACL). ACLs can grant access privileges explicitly to users, groups, or roles.

The Repository provides the following basic level of permissions:

- None – Objects with users having “None” permission cannot see the objects in search results or through folder navigation. If you don’t want a user or group to view your assets, give “None” permission to that group or individual user in ACL.
- Browse – This permission allows users to view the metadata but not the content. User can search for this document, the system will show the document metadata in search results, but users cannot open the content.
- Read – This permission allows users to read/view the metadata and content of the asset.
- Relate – This permission allows user to read/view metadata and content and it also allows users to create relations. This is useful when a user is trying to create annotations on the assets.
- Version – This permission allows users to create new versions of the asset. The system will not allow the user to save an updated document as the same version. The User also gets Read and Relate permission.
- Write – This permission allows the user to create new documents and update the old document. The System allows users to save the updated old document as the same or a new version. The User also gets Read, Relate, and Version permissions.
- Delete – The User has right to Delete the assets along with Write, Version, Relate, and Read permissions
- DELTE OBJECT – This is an additional permission granting users the ability to delete the asset. It will take away other permissions that are allowed in regular delete as described in the above bullet.

Each of the above permissions is hierarchical. For example, if the user has write permission on a file then he/she also has version, relate, read, and browse permissions.

The Repository also provides extended permission sets which are also part of ACLs. These permissions are not hierarchical.

- Change Location – Allows user to move objects from one folder to another
- Change Ownership – User can change the owner of the object
- Change Permission – User can change the basic permission on the object
- Change State – User can change the document lifecycle state
- Execute Procedure - User is allowed to execute an external procedure associated with object
- Change Folder Link – User allowed to Link an object to a folder or Unlink a object from a folder

Trusted Content Services (TCS)

TCS is an add-on service that requires a license. It allows the following additional functionality in the repository:

- Content encryption
- Digital Shredding of Content
- Multi Dimensional Access Controls (MDAC)
- Digital Signature

I will address the first two topics in this article. Please refer to the Documentum Content Server Administration guide for complete details on the remaining two topics.

Content Encryption

If you create an encrypted file store in the repository, all the content saved to this file store is encrypted before writing to disk. When a user reads this content, the repository decrypts the file in the content server before forwarding the file/document to user. This feature protects the data stored on the disk so that the system administrator or a storage administrator cannot read the content from the operating system. There will be a 5 to 10% performance hit depending on the file size when you encrypt the content.

Applications can be designed to store only highly sensitive documents in encrypted file store and the balance of documents in non-encrypted (regular) file stores. You can also

have multiple encrypted file stores and configure an application to segregate the content based on the document type.

Content in an encrypted file store is encrypted using a symmetric key (called FEK or File Encryption Key). This key is unique for every content object. The FEK is encrypted using another key called FSK (or File Store Key). This key is different for every file store. It is stored in the "dm_filestore" object associated with the encrypted file store. The FSK is encrypted using a DBK (or Docbase Key), and the DBK itself is encrypted using the AEK (or Application Encryption Key). The DBK is unique for every repository; it is stored in docbase config object "dm_docabase_config" The AEK is the master key for the system. AEK key is stored on the Content Server Installation location ($DOCUMENTUM/dba/secure) as shown in Figure 1 (Documentum Encryption). All key lengths are 192 bits and 3DES-EDE-CBC is the algorithm employed for each encryption.

The DBK and FSK are Base64 encoded after encryption and stored in the database. The encrypted FEK is stored along with the content as part of the object properties. The AEK is encrypted using a passphrase using PKCS #5 standards. This is the passphrase you need to enter when the Content Server is being started. The algorithm used for AEK encryption is 3DES-EDE-CBC. The iteration count is 1024, and the salt size is 8 bytes. The encrypted AEK is stored in a file.

During installation, the system creates the AEK key with a default passphrase. I recommend changing the passphrase before configuring the first repository.
I also recommend taking a backup of the AEK key.

Content File Stores on storage have full permissions to the Content Server Installation owner. No access is granted on Files Stores at the OS level to any user or group outside the Installation Owner.

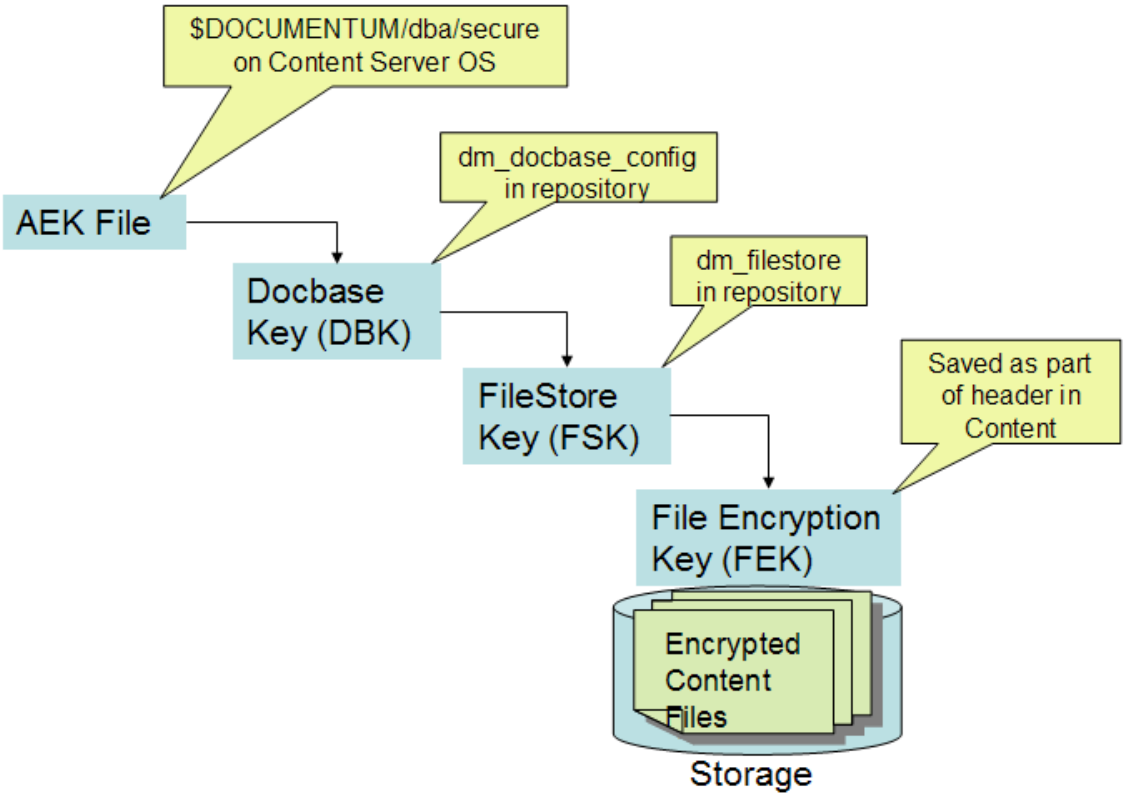

Figure 1 (Documentum Encryption)

Digital Shredding of Content

When a user deletes a document from a repository, the system deletes the sysobject but leaves the content file on storage/disk. This is called orphan content. Documentum has a utility called dmfilescan that generates a script with a list of all the orphan files on the disk for all the file stores. The Documentum administrator must run the generated script to remove the orphan content from the file system. There is a time gap from the deletion of an object in the repository to the deletion of a physical file on the disk.

Digital Shredding can be turned on at the file store level. When this option is on, the system will delete the content on disk at the same time as the user deletes the document in the repository. This is done in a single transaction. This option is frequently used in compliance environments.

Digital shredding provides a final, complete way to remove content from a storage area by ensuring that deleted content files may not be recovered by any means. The system automatically writes over the location data multiple times to ensure that the file cannot be recovered by disk utilities or other security software.

Auditing

Auditing is the process of recording the occurrence of system and application events in the Repository . Events are operations performed on objects in a Repository or something that happens in an application. System events are events that the Content Server recognizes and can audit. Application events are user-defined events. They are not recognized by the Content Server and must be audited by an application. By default, the Content Server always audits the following system events:

1. All executions of an Audit or Unaudit method
2. User login failure
3. All executions of a Signoff, Adddigsignature, or Addesignature method
4. You can also audit many other operations. For example, you can audit:
    - All occurrences of an event on a particular object or object type
    - All occurrences of a particular event, regardless of the object to which it occurs
    - All workflow-related events
    - All occurrences of a particular workflow event for all workflows started from a given process definition
    - All executions of a particular job

Please refer to Content Server Administration guide for complete list of audit events.

Single Sign on (SSO)

Documentum, out of the box, supports Single Sign On with Netegrity SiteMinder and RSA Access Manager. Let's look at more detail of the SiteMinder solution.

Netegrity SiteMinder offers SSO functionality across single and multiple domains, simplifying the use of applications across various Web and application servers and across multiple operating systems. It also provides policy-based, centralized control of user authentication and access management. The SiteMinder Web agent is installed on the WebTop application Server. This agent will check all traffic to the application server for SiteMinder trusted user session cookie as part of the header. If the agent detects an un-trusted session or no SiteMinder cookie in the header, it will forward the user to the SiteMinder server for authentication before continuing to process the user request.

The Netegrity configuration performs the following operations:

- Netegrity invokes an authentication page when the user accesses Documentum WebTop through the proxy after the Netegrity setup.
- The user can then select a repository and log in to Documentum WebTop without providing login credentials.

Execute the following steps when a user tries to access the WebTop application (which is SiteMinder protected as shown in Figure 2 (Documentum SSO)). If the SiteMinder is configured for certificate based authentication, it will automatically validate user credentials using the certificate from the users desktop. If you configure SiteMinder for form based authentication, the user is prompted with a login screen to enter userid and password for SiteMinder.

1. User invokes WebTop URL from browser
2. The SiteMinder web agent intercepts the request and redirects the user to the SiteMinder Policy Server to get authenticated
3. The User browser connects to the SiteMinder Web server to get authenticated
4. Based on the authentication mechanism (form based or certificated based), the user gets authenticated with the SiteMinder Policy Server. The Policy server generates a ticket and stores it in the SiteMinder cookie.

5. The SiteMinder web server redirects the browser to the WebTop application to continue with the user request. The User browser session has SiteMinder cookie with valid ticket
6. The User browser resends the request back to WebTop with the SiteMinder cookie
7. The SiteMinder web agent checks the cookie and validates the ticket with the Site Minder policy server. Once the web agent confirms the validity of the ticket, the request is passed to the WebTop application.
8. The WebTop application is configured to extract the SiteMinder cookie information and automatically passes it to the Content Server for authentication.
9. The Content Server takes the SiteMinder ticket from the WebTop request and validates it with the Policy server.
10. If the ticket is valid, the Content Server creates a user session and returns to WebTop.
11. WebTop returns the request to the user with a valid user session created.

See illustration on the following page.

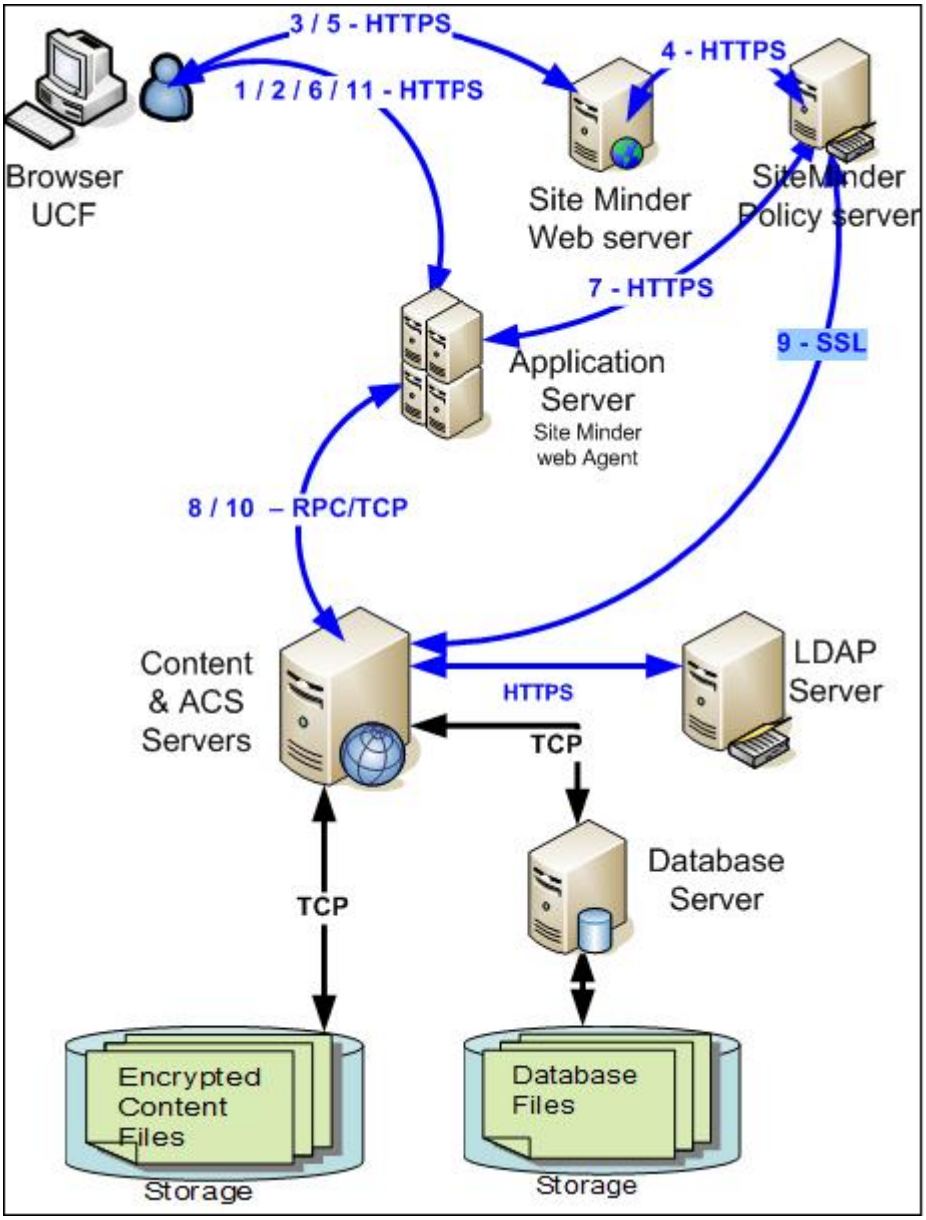

Figure 2 (Documentum SSO)

Secure Communication

You can configure the Documentum Infrastructure for a secure communication from end-to-end as shown in Figure 3 (Documentum Repository). The Content server can be configured to listen on a secure port, an unsecured port, or both. Based on your business requirements, you can turn configure the repository to any of the three options.

In WDK or custom DFC applications, configure the dfc.properties file to connect to the content server with a secure or unsecure port. All client programs try to connect first to the unsecure port by default, if that port is not available the client will try to connect to a secure port.

If you have turned on SSL / HTTPS between user the browser and application server, deploy a certificate generated by a trusted CA vendor so that Java on the users' desktops has a trusted CA authority in the local keystore. If you are using a private internal certificate, make sure you deploy the CA on users' java keystore.

The Content Server uses the following SSL specifications to communicate with DFC (WDK) clients on a secure port.

- ADH (Anonymous Diffie-Hellman) for key exchange
- key size: 1024 bits
- AES for data encryption; key length: 256 bits
- Hashing algorithm: SHA-1 (Secure Hashing Algorithm)

See illustration on the following page.

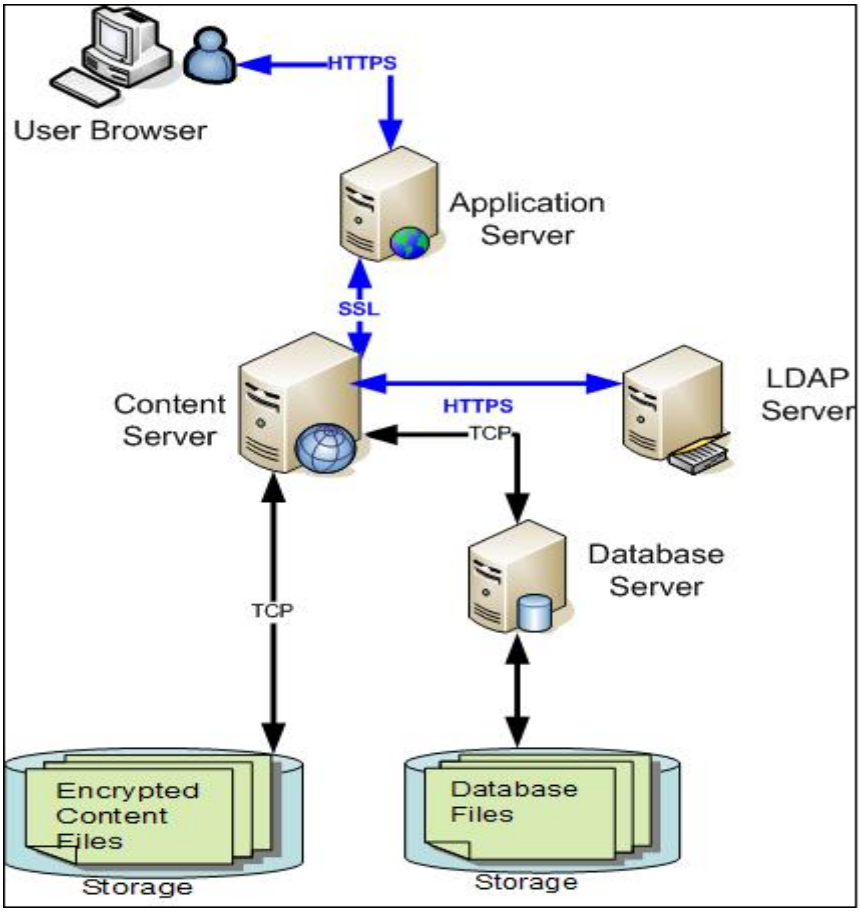

Figure 3 (Documentum Repository)

Documentum Content Read Transaction

Let's look into Documentum Read/View Transaction as shown in Figure 4 (Documentum Secure Transactions). The user will open a document from the repository in this transaction.

Documentum applications install a Java UCF applet on the users' desktop. This applet is invoked when users execute any content transactions such as import, export, view, checkout and check-in a document from the user desktop. UCF connects to Accelerate Content Server (ACS) to import or export content to the repository. The ACS server runs on content server embedded JBoss application server. Communication is HTTP or HTTPS between UCF and ACS.

The following steps are executed for File View operation:

1. Browser sends View operation request to Application Server
2. Application server sends the View request to Content server using DFC RPC call
3. Content Server checks whether the user has a view permission on the document
4. If the user has the correct permission, the content Server sends the request back to the application server with instructions to build ACS URL to access the document. These instructions are digitally signed by the Content Server using a private key stored in the dm_cryptographic_key object in the repository
5. Application server builds the ACS URL to fetch the document using UCF and forwards the URL to the user
6. User browser invokes UCF applet and passes the ACS URL to fetch the content. UCF directly connects to ACS server running on Content server embedded JBoss to fetch the document.
7. ACS server first validates the signature on the request, using the public key provided to the server. There is one public key per repository, stored in dm_public_key object.
8. ACS fetches the document from storage upon successful request validation
9. ACS forwards the document to the UCF client applet
10. UCF invokes appropriate viewer to open the document on the users' desktop

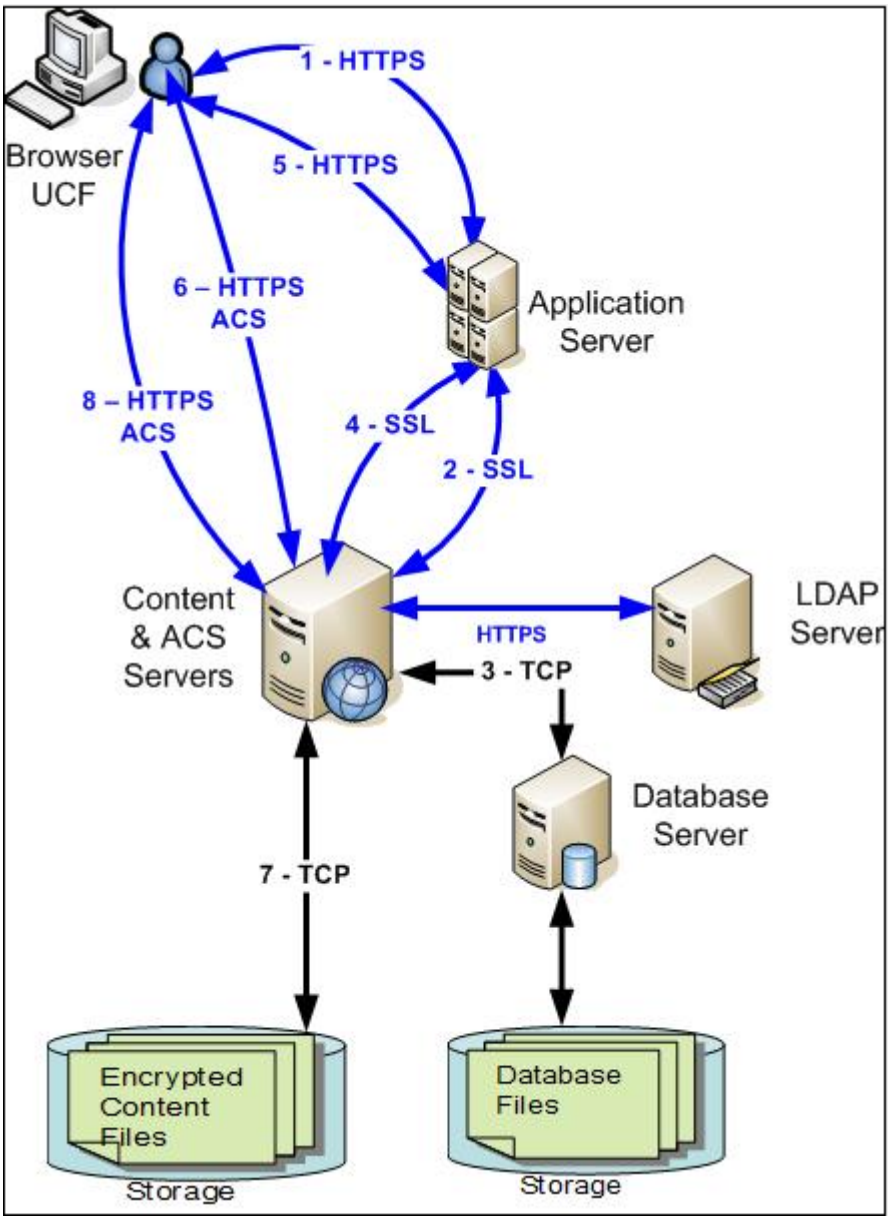

Figure 4 (Documentum Secure Transactions)

In the above example, we have all the communications with the Applications server and the ACS and Content Server are secure. This will have a performance impact on users. If your business requires only the content transfer communication to be secured and the balance of transactions can be unsecured, you can configure HTTPS communication only between user browsers to ACS server as shown in Figure 5 (Documentum Secured Content Communication).

This configuration will improve the performance when user navigates through folders, searches for documents, views metadata etc. When the user executes any content transfer operations like viewing documents, check-out, check-in, export and import then the secure HTTPS communication is used.

You need to perform following steps to configure the ACS for HTTPS

- Configure ACS server to accept HTTPS requests
- Update dm_acs_config object in repository, set acs_supported_protocol and acs_base_url attributes
- Restart ACS server

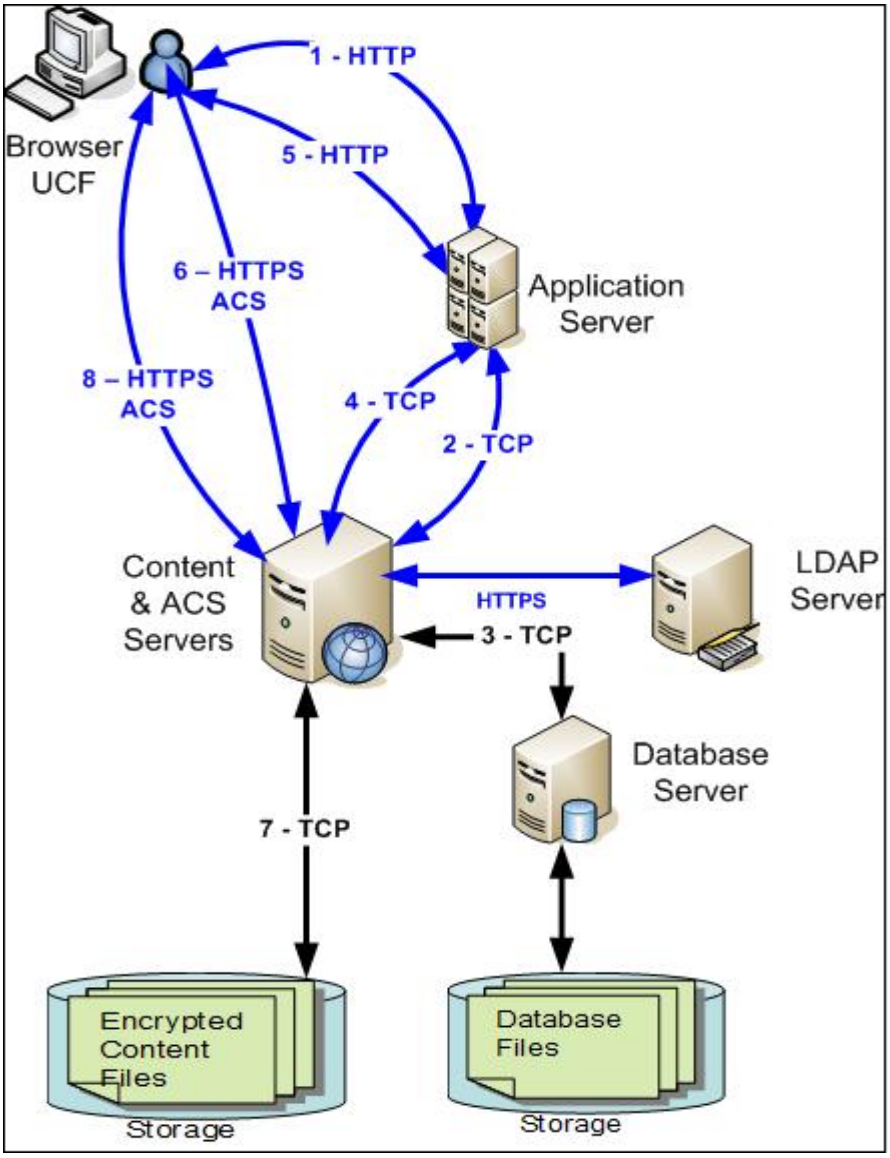

Figure 5 (Documentum Secured Content Communication)

SSL Offloading

You can gain performance by offloading the SSL at Load Balancer(LB) / proxy server as shown in Figure 6 (Offloading SSL) if Documentum Content Servers and WebTop application servers are deployed in a secure network zone. In this configuration, the servers behind the LB / proxy server are not reachable to end users directly. The traffic between the user browser and load balancer/proxy server is HTTPS and all the communication gets encrypted / decrypted in the LB / Proxy server. Users will see better performance with this configuration.

Proxy ACS requests

All Content request operations need direct communication between the user Java process (UCF applet) with ACS server running on the Content Server host. If you don't like the idea of user JVM directly connecting to the Content Server host, proxy the requests between UCF and ACS server, create a virtual URL , and map the virtual URL to actual ACS host in proxy server. Update the "dm_acs_conig" repository object, set acs_base_url attributes to virtual host name.

Example: If you have ACS running on host http://cs1.emc.com:9080/acs, create a virtual host (https://acs1.emc.com/acs) and proxy the request to http://cs1.emc.com:9080/acs. In this example, we are Offloading SSL in LB / proxy server to gain performance. This configuration will address two issues: SSL Offloading and restricting direct access from User browser to Content Server host as shown in Figure 6 (Offloading SSL).

See illustration on the following page.

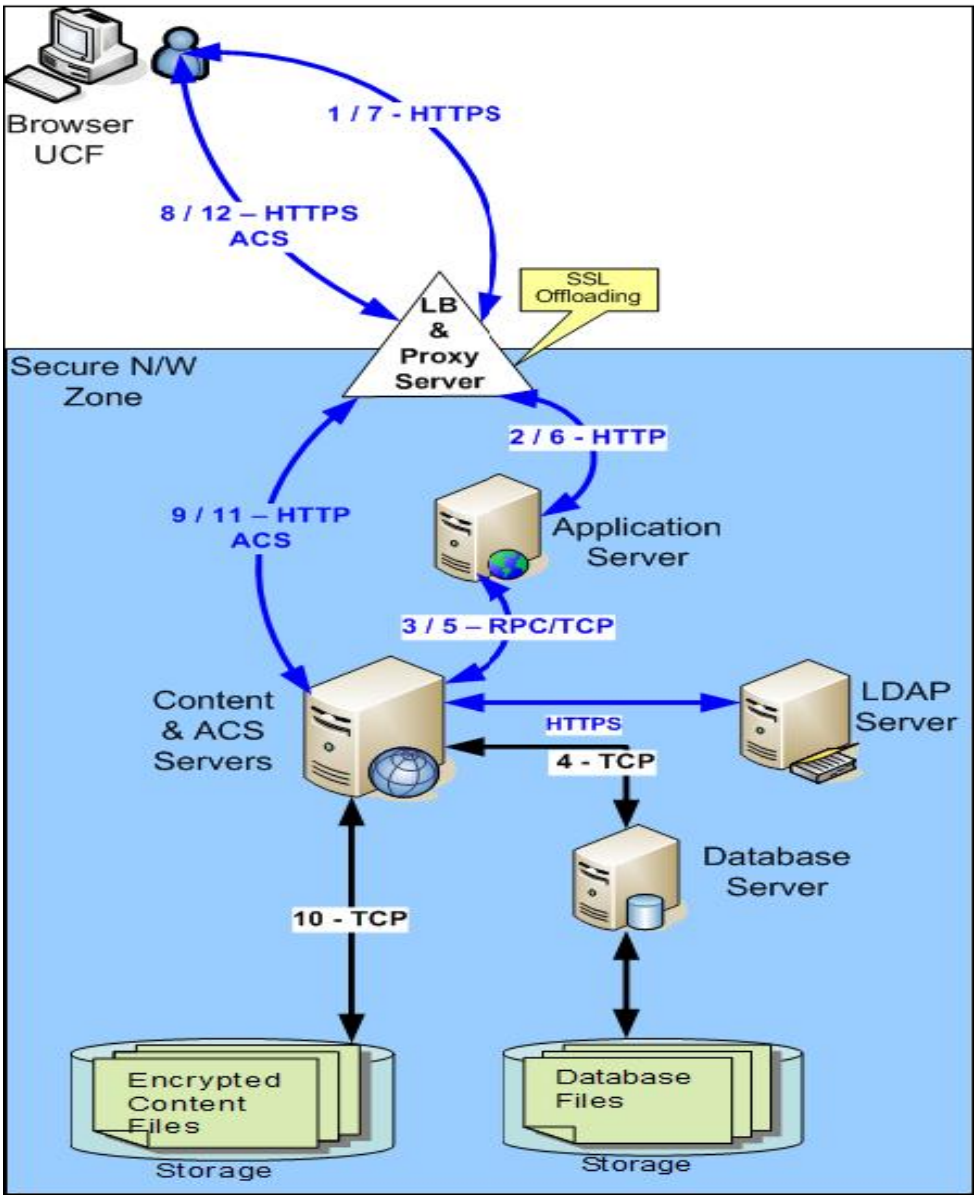

Figure 6 (Offloading SSL)

Approved DFC Clients

By default, all the DFC clients (WDK or custom client) are allowed to access the repositories if the client user session has right privileges. For example, if your business requires users to access the repository using a particular client (Custom DFC or WDK client), you can configure the repository for approved client app requests so that the repository would only entertain requests from pre-approved clients.

The following steps will help Documentum administrators turn on Privileges clients:

1. Login to DA as a super user
2. Navigate to privileged client administrator node
3. Click Managed clients button
4. Add the entry of the associated client to the privileged client list
5. Turn ON approved client option at docbase level “dm_docbase_config” repository object. Set “approved_clients_only” property to true, restart the repository.
6. Try to connect to the repository using the WDK or DFC client
7. Repository will not allow user to create a session if the client is not in the privileged client list.

Documentum Administration Concerns

The Repository Installation owner is a “superuser” by default. By virtue of being a super user, the installation owner has the highest privileges in the repositories and can read, change, and delete permissions on any object within the repository. A superuser is eligible to perform all the administration tasks including assigning the superuser role to other users in the repository. You can assign the superuser role to members of your Documentum support / administrator team to carry out regular administrative tasks. All the actions performed by these super users can be audited.

Business users always have concerns about unauthorized access to sensitive corporate data by in-house users/admins. There are a few ways to mitigate this risk:

- Turn on auditing for all actions/operations (view, export, check-in, check-out etc) related to documents. You can periodically audit the audit logs for any misuse of the repository by administrators.
- Most of the Repository administration tasks can be performed by user with Sysadmin role, this role is a one step down from the superuser role. Sysadmins can perform most of the superusers' tasks with no access to documents / objects. You can assign sysadmin role to support team for system maintenance.

You need the installation owner account to start/stop Documentum application. You can also use this account to login to DA client for system maintenance. As multiple people in the administrator team share the same installation userid and password, it will be difficult to audit who has executed a particular operation.

The following guidelines will mitigate the risk associated with using the Installation Owner account to login to Content Servers running on a UNIX system:

- Use the Two-Factor authentication model for admin team members to login to Content Server
- Do not allow direct SSH or telnet to content server from user desktop
- Allow access to Content Servers through a secure gateway server
- Admin user must authenticate using an RSA ID on a gateway server before SSH into content server
- Once logged into the Content Server with an individual ID, the person should SUDO as installation owner
- As Installation owner user can stop/start and do other admin operation in the repository
- The above steps will create a trace of audits in UNIX system

The following guidelines mitigate the risk associated with using the Installation Owner account for login to the repository from Documentum Web Clients like WebTop or DA:

- Protect WebTop & DA web clients with SiteMinder or RSA
- Make sure that the Documentum Installation Owner account is not part of SiteMinder or RSA policy server
- Administrators cannot use the Installation account to login to DA or WebTop, this will force administrators to use individual IDs for system maintenance
- Assign sysadmin or superuser role to individual admin team members

Content Server Installation Owner Password Rotation

The system allows changing the installation owner password. Shutdown all the services before changing the password for a Content Servers running on UNIX. On Windows machines, change the password in the service entry for repositories before starting the repository.

Information Rights Management (IRM)

Most of the Enterprise Content Management systems like to control how corporate documents are viewed, printed, downloaded, copied etc. The Documentum Repository only provides document security within the repository. Once the document has left the repository when a user downloads, views, or copies it onto his/her desktop, there is no control on these documents. You need to install Documentum IRM application if your project requires document control after leaving the repository. Documentum IRM server with client pug-ins provides a strong security based on the policies defined in the repository to protect the documents after they leave the repositories.