A Security Analysis of the OpenText Content Server Electronic Document Management System

Bc. Petr Ježek

Master’s thesis
2021

Tomas Bata University in Zlín
Faculty of Applied Informatics

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav elektroniky a měření

Akademický rok: 2020/2021

ZADÁNÍ DIPLOMOVÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Bc. Petr Ježek**
Osobní číslo:	**A18569**
Studijní program:	**N3902 Inženýrská informatika**
Studijní obor:	**Bezpečnostní technologie, systémy a management**
Forma studia:	**Kombinovaná**
Téma práce:	**Bezpečnostní analýza systému pro správu elektronických dokumentů OpenText Content Server**
Téma práce anglicky:	**A Security Analysis of the OpenText Content Server Electronic Document Management System**

Zásady pro vypracování

1. Carry out a literary search on the topic of the work.
2. Describe the document management system and its typical configuration.
3. Describe and categorize possible threats and attack vectors.
4. Design methods for security analysis of potential attack vectors.
5. Evaluate the level of risk of the identified threats.
6. Design appropriate solutions to eliminate threats.
7. Evaluate the chosen solution.

Forma zpracování diplomové práce: **Tištěná/elektronická**

Seznam doporučené literatury:

1. HENRIQUES DE GUSMÃO, Ana Paula, MENDONÇA SILVA, Maisa, POLETO, Thiago, CAMARA E SILVA, Lúcio and CABRAL SEIXAS COSTA, Ana Paula, 2018. Cybersecurity risk analysis model using fault tree analysis and fuzzy decision theory. International Journal of Information Management. 1 December 2018. Vol. 43, p. 248&#x2013;260. DOI 10.1016/j.ijinfomgt.2018.08.008.
2. HUBBARD, Douglas W., SEIERSEN, Richard, GEER, Daniel E. and MCCLURE, Stuart, 2016. How to Measure Anything in Cybersecurity Risk. 1st edition. Wiley.
3. KENNEDY, David, O&#x2019;GORMAN, Jim, KEARNS, Devon and AHARONI, Mati, 2011. Metasploit: The Penetration Tester&#x2019;s Guide. 1st edition. No Starch Press.
4. REGALADO, Daniel, HARRIS, Shon, HARPER, Allen, EAGLE, Chris, NESS, Jonathan, SPASOJEVIC, Branko, LINN, Ryan and SIMS, Stephen, 2018. Gray Hat Hacking: The Ethical Hacker&#x2019;s Handbook, Fifth Edition. 5th edition. McGraw-Hill Education.
5. WEAR, Sunny, 2018. Burp Suite Cookbook: Practical recipes to help you master web penetration testing with Burp Suite. 1st edition. Packt Publishing.

Vedoucí diplomové práce:	**prof. Mgr. Roman Jašek, Ph.D.** Ústav informatiky a umělé inteligence
Datum zadání diplomové práce:	**15. ledna 2021**
Termín odevzdání diplomové práce:	**17. května 2021**

UNIVERZITA TOMÁŠE BATI VE ZLÍNĚ
Fakulta aplikované informatiky
DĚKAN

L.S.

doc. Mgr. Milan Adámek, Ph.D. v.r.
děkan

Ing. Milan Navrátil, Ph.D. v.r.
ředitel ústavu

Ve Zlíně dne 15. ledna 2021

ABSTRAKT

Tato bezpečnostní analýza systému pro správu elektronických dokumentů OpenText Content Server představuje přehled možných zranitelností a jejich řešení. Záměrem bylo představit jednotlivé části systému, jejich role a vliv na bezpečnost systému. A na základě těchto informací identifikovat možné zranitelnosti. Cílem bylo prozkoumat v praxi často přehlížené nebo starší části systému, jako jsou výchozí nastavení a integrované sofwarové komponenty třetích stran. Identifikované zranitelnosti, vektory útoku, taktika a techniky byly klasifikovány pomocí standardů MITRE. Výsledkem této práce je ověřený postup jak systém dodatečně zabezpečit.

Klíčová slova: OpenText, Content Server, Archive Server, Archive Center, Livelink, bezpečnostní analýza, kyberbezpečnost, podnikový

ABSTRACT

This Security Analysis of the OpenText Content Server Electronic Document Management System provides an overview of possible weaknesses and creates a hardening guide for their mitigation. It aims to develop a better understanding of the system components, their functional role, and how they affect system security. Such an understanding helps to recognize possible weaknesses. The focus was mainly on overlooked or legacy parts of the platform, such as default settings and integrated third-party software components. Identified weaknesses, attack vectors, adversary tactics, and techniques were classified using MITRE standards. The hardening guide with tested steps is created as an outcome of this thesis.

Keywords: OpenText, Content Server, Archive Server, Archive Center, Livelink, security analysis, safety analysis, cybersecurity, enterprise

To my beloved wife Marketa, who made the creation of this thesis possible.

TABLE OF CONTENTS

INTRODUCTION

Is my system secure?

That is quite a burning question for many IT professionals and managers today. As the number of cyber-attacks on big companies is growing and getting into mainstream news media, managers finally realized that underinvesting in cybersecurity can lead to remarkably high clean-up costs afterward. Moreover, there is no need to search very deeply to see instances of such threats to happen to even very well-established companies, for instance, Maersk or TNT Express (FedEx).

It does not mean that there were no investments into cybersecurity before. They were mainly directed to technologies separating wild Internet from safe Intranets. Nowadays, we can see the shift to a zero-trust security model. There is an understanding that not only external threats are costing us money. Internal threats can cost us, too. It does not make a big difference if it is an employee who steals our valuable information, or it is an attacker who gets into our "safe" network and is gathering valuable information for months. The loss at the end is similar. Companies are more often hiring security experts as internal employees. They started PEN testing their core information systems, which sometimes can be a decade old. These systems are often without the latest updates because they "do not touch a running system."

We do not learn too much about attacks, where an employee steals valuable information and sells it to competitors or enemy states. These types of attacks are not so often made public. It is easier to hide them because they do not halt a whole company. Receivers of stolen information do not advertise the fact that they stole it. Often, just a few people within a company are aware. Furthermore, it is embarrassing. The primary targets of these attacks are information.

This work does not focus on preventing malware attacks as described at the beginning of this chapter. However, some of the improvements described later may also prevent an attacker from getting a foothold into a corporate network. This thesis aims to create an initial version of a more holistic approach, which may help prevent targeted attacks. It does not matter, whether external or internal.

Legacy systems, legacy issues

Much work has been done to secure software by design. Responsible vendors are security testing their products using static or dynamic application security testing tools. Additionally, using state of the art tools and techniques to discover possible issues before their product is shipped to customers. However, with enterprise software solutions, this approach is not sufficient. Complex systems may be built by integrating many different components. To use an analogy: A brick factory may create perfect quality bricks, but a bricklayer may build a dangerous wall from those great bricks. It depends on how the bricks are layered; also, what is used to cement them matters as well. Therefore, the attack vector of complex systems is not a simple sum of all integrated components. Figuratively, the resulting attack vector of those integrated systems is a multiplication of all components instead.

The Content Server platform is often used in such an integration. OpenText builds the platform itself, but it is often used as a central solution for many connected business applications from different vendors. In some scenarios, some of those systems are built for years before they are used in production. In others, they are built incrementally over the years. The new parts are connected one by one to the legacy systems. These incremental integrations lead to situations when an organization has no one knowing the whole system. While building such a system, organizations focus on functionality and costs. Often security is considered only during the design phase, where the architecture or permission model is designed. Rarely, there is further focus on hardening the whole solution.

This work aims to look into dark places, often overlooked during the implementation process: integrated known components and default settings. As the security of commonly used parts like operating system, database, firewall proxy, and built-in security settings of the application itself are usually well documented, they will not be covered in this thesis. The result is the first version of a supplemental hardening guide. Hopefully, this first version will lead to a comprehensive hardening guide for the Content Server platform in the future.

The idea that software solutions with many features or complex way implemented features may be more prone to security issues than simpler ones was intriguing. Uninstalling unnecessary software, disabling not used features seemed contrary to the traditional approach in the industry. Researching the effect of an attack surface size on security while working on this thesis confirmed this approach. Additionally, it confirmed that taking into account functionality available only internally during the security analysis is a valid approach.

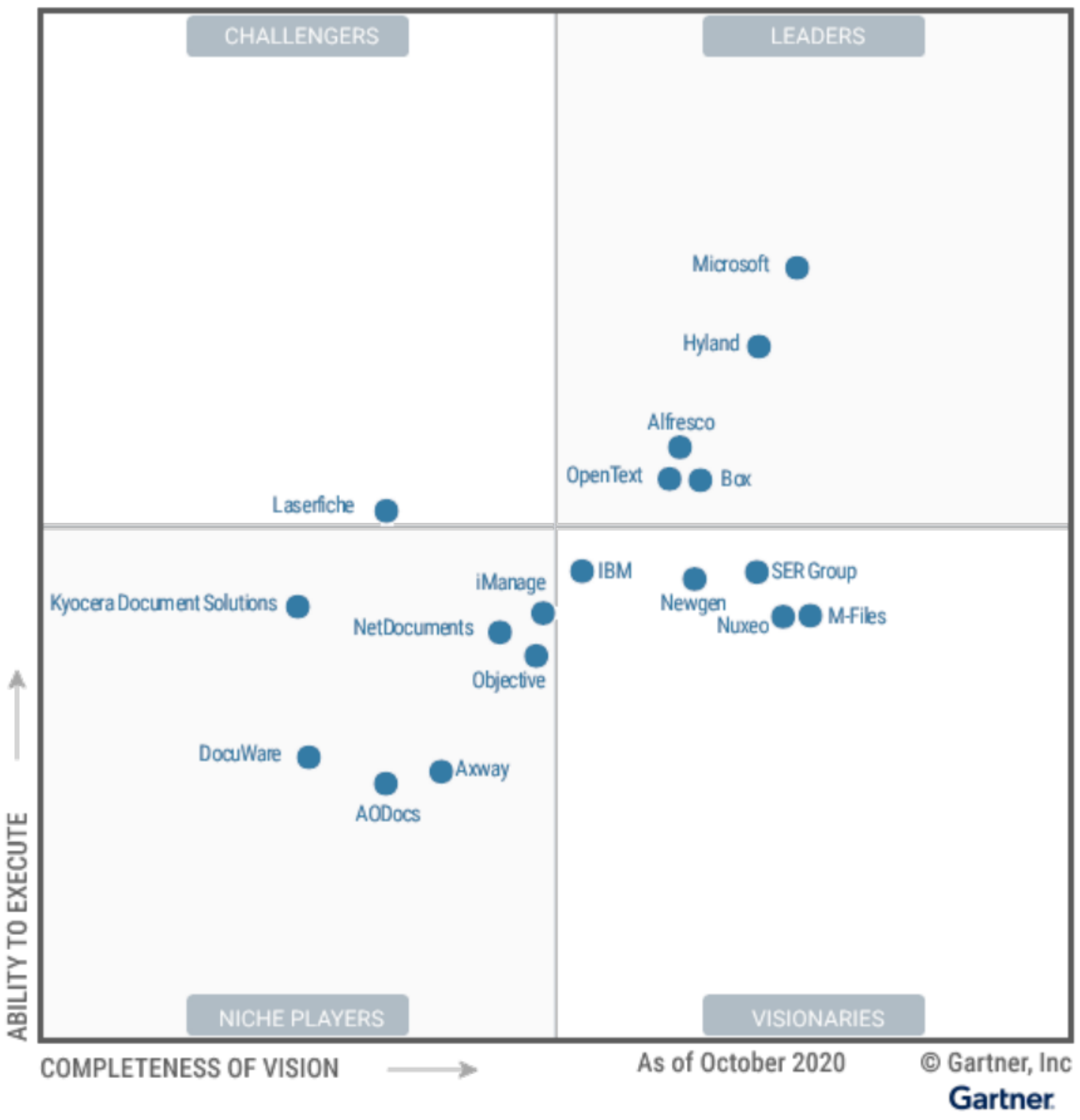


Figure 0.1 Magic Quadrant for Content Services Platforms[1]

Why should we care?

OpenText has the most significant market share in the Content Services Platforms market globally. The primary customer verticals of OpenText are financial services, public sector, energy, and utilities. In addition, the Content Server platform is also used by some army branches, weapons research, regulators of nuclear energy, nuclear power plants, and hospitals.

OpenText has a close partnership with SAP and a robust ecosystem of over 600 international partners implementing and supporting multinational clients.[1]

The thesis is based on more than two decades of experience building these systems and seeing many shortcomings with possible security consequences, including a yearlong project restoring a system wipeout in one of the significant cyber-attacks and handling other consequences of this attack. Furthermore, this work may bring new talking points for system owners, why it is crucial to invest in additional security improvements of their systems.

The reader looking only for information on how to harden the Content Server platform may go directly to the chapter 5. However, if also interested in information about possible weaknesses of the platform, may jump to the chapter 3.

The reader with the OpenText Content Server platform experience can skip chapter 2

I. THEORY

1 LITERATURE REVIEW

The goal of this literature review was to get enough background info to answer the following questions:

- How is a security analysis of complex enterprise systems done? Is there available info on the security of the Content Server platform?
- How is an attack surface defined, and how should it be used in this thesis?
- How to evaluate risks found during this analysis?

1.1 Attack Surface Analysis

"Intuitively, a system's attack surface is the set of ways in which an adversary can enter the system and potentially cause damage." Is the most used definition of attack surface according to Theisen et al. in *Attack surface definitions: A systematic literature review.*[2]

The same source also states: "While the phrase attack surface is used in a variety of contexts in cybersecurity, professionals have different conceptions of what the phrase means."[2]

Theisen et al. define attack surface in the following six contexts:

- **Barriers** – the method of preventing attacks, rather than the paths attacks can occur on, by malicious parties.
- **Methods** – the methods of implementation, data channels, and data present in the system, with no specific attack features mentioned.
- **Flows** – attack surface is defined as a data flow and control flow only, without considering methods or avenues of attacks.
- **Reachable Vulnerabilities** – the vulnerabilities exposed to end-users via paths or flows rather than the paths or flows themselves.
- **Adversaries** – attack surface is the union of all possible ways an attacker could cause damage to a system.

- **Features** – the attack surface is an enumeration of all available attack avenues to a target system.[2]

Looking at those definitions in the context of this work:

- **Methods** – While deployment and customer-specific implementation may also be understood as implementation, this thesis uses it in the system's development context.
- **Reachable Vulnerabilities** – This context of attack surface definition would limit us to investigate only possible external threats. For example, parts of the platform available only to administrators would not be considered in this context as they are not externally accessible. However, internal threats do not bear negligible risk in knowledge workers' corporate environments, and they are discussed in this work.
- **Adversaries** – The focus is on points in the system with active attacks. This definition of attack surface is too limiting for this work.
- **Flows** – This approach may help significantly with uncovering undetermined interactions and transient data in complex systems. In addition, analysis of the Content Server platform using, for example, *Threagile, Agile Threat Modeling Toolkit*[3] can bring exciting findings. However, this thesis focuses on practical use, and changing the Content Server code or other components is not an option.
- **Barriers** – Hardening of the Content Server platform as prevention of attacks is part of this thesis. However, it is not the only focus, and not all components hardening will be addressed here. Some of the components of the platform are pretty standard components. There is a lot of hardening guides available for them. They are generally properly hardened during deployment of the system, like Operating Systems, RDBMS, firewalls, proxies. Moreover, this was the reason why this thesis does not cover them. The only exceptions are application servers, as those are deployed mainly by consultants installing the system. Furthermore, they are usually used without further hardening.
- **Features** – The context of features as a list of all possible attacks to the system aligns with this work's aim. It requires an enumeration of system components, which are possibly susceptible to an attack. For example, in case our Content Server platform uses only five components, which has fewer open ports, machine-to-machine communication, published web services, third party solutions, and

features compared to a similar setup with ten components. As a result, the later version of the platform will have a larger attack surface than the former.

This work aims to create a broad list of possible security issues for further investigation in the future. In addition, it should illustrate the complexity of securing the Content Server platform. Furthermore, an essential hardening guide using configuration changes for easily mitigated attack vectors.

The work will use the phrase attack surface in the context of the whole system. However, as mentioned before, the system's parts commonly covered by hardening guides, installed and configured by internal IT staff (like Operating Systems, RDBMS, firewalls, proxies) are excluded.

Using features context for attack surface definition allows us to ignore the attributes, which would typically influence the size of the attack surface as-is:

- users count
- value of stored data
- customer visibility
- customer's controversy or hate towards customer

This work aims to be as general as possible. It uses a typical platform instead of a typical customer. Moreover, including the attributes mentioned above would add unnecessary complexity to this work.

The original aim of this research was to use one of the more systematic approaches for the analysis of weak points of the platform. However, the list of possible weaknesses created during initial brainstorming was so long that using a formal method for discovering additional weaknesses was unnecessary.

1.2 Security Analysis of Enterprise Systems

It would be beneficial to design this analysis using existing research on this topic. However, this was not done because there is seemingly — based on the attempted search — a lack of research on security analysis of enterprise systems with a similar focus. Instead, the vast majority of available enterprise systems analysis is based on source code analysis or analysis of flows.

The work *Measuring the Attack Surfaces of SAP Business Applications* by Pratyusa K. Manadhata et al. [4] also analyzes source code. The resulting Attack surface is based on a count of entry and exit points in Java code. Interestingly Manadhata et al. compared three different versions of a service in different versions of the platform. Based on the resulting measurement, they were able to compare which solution has a lower attack surface. Manadhata et al.'s recommendation for SAP customers: "Software consumers often have to make a choice between several possible configurations of software. For example, SAP business applications can be configured in many different ways; SAP customers choose the configuration best for them. Configuring large enterprize-scale software is a complex process; hence choosing an appropriate configuration is a non-trivial and error-prone task. SAP's customers could use a system's attack surface measurement as a guide in choosing an appropriate configuration. Since a system's attack surface measurement is dependent on the system's configuration, they would choose a configuration that results in a smaller attack surface exposure."[4]

1.2.1 Open and Academic Information on Security of the Content Server

After filtering search results and removing old versions of the Content Server (also known as Livelink – version 9.7.1 released in December 2007 and sustaining maintenance in September 2013 [5]), different OpenText products, different topics (search, Knowledge Management, Enterprise Content Management, web mining, configuration of competitors product), limiting publishing date from the year 2000 onwards and language to English, there are no relevant documents on Content Server platform security.

1.2.2 OpenText Information on Security of the Content Server

OpenText addresses security of the Content Server platform in several documents and Knowledge Base (KB) articles relevant to this work:

- *Best Practices Content Server Application Security Hardening Guide v10.5 and 16.pdf* [6]
- *Security Capabilities Overview for OpenText Content Suite.pdf* [7]
- *OTDS - OpenDJ Backend Security Hardening of Ports and Certificates* [8]

- *KB article: Content Server - OTDS - System Center - Tempo Box - How to implement custom generated Certificates for OpenText products* [9]
- *Archive Server Encryption of Documents* [10]
- *Secure Archiving* [11]

Documents mentioned above are available to partners and OpenText customers via OpenText My Support portal. However, these are not publicly available.

These documents are mainly focused on the security settings of the product. This work is intended to extend it by looking into the security of the whole platform.

1.3 Risk Evaluation

The most common approach for measuring the risks associated with vulnerabilities in a cybersecurity area is the Common Vulnerability Scoring System (CVSS). It is an open framework for communicating the characteristics and severity of software vulnerabilities. [12]

However, its results can vary in case of incomplete information. Furthermore, it assumes that vulnerability was discovered and verified. Therefore, not all weaknesses found during the attack surface analysis in this work could be understood as vulnerability. Instead, some of them may be better described as exposure. [13]

Using Common Weakness Scoring System (CWSS™) is more appropriate in this context as it considers design weaknesses. CWSS scoring supports the use of incomplete information. Additionally, it takes into accounts if a weakness can be misused internally or externally. [14]

CWSS is designed to measure the severity of weakness. It does not calculate the risk of and weakness. Proper risk evaluation for each weakness found will require incorporating attributes mentioned at the end of chapter 1.1, like user count, a value of stored data, and others. For the measuring risk score of the weaknesses described in this work, the use of CWSS will be sufficient.

For faster calculation of CWSS score, the online calculator is used. [15]

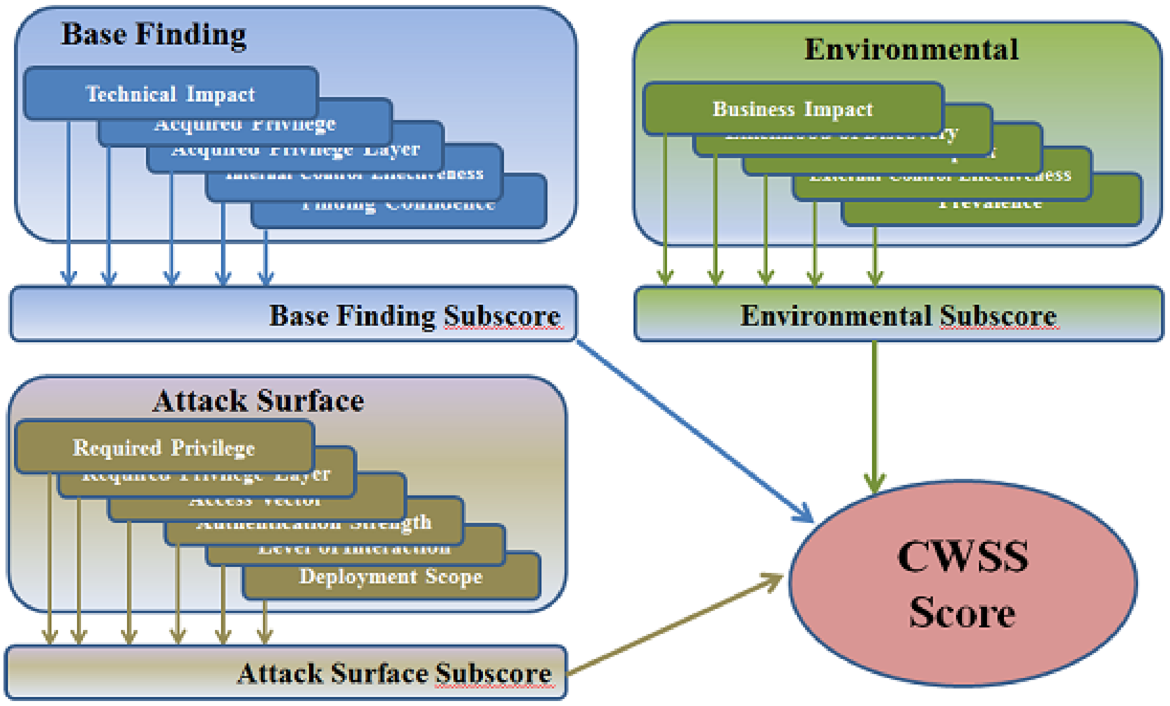


Figure 1.1 CWSS Score[14]

1.4 Attack Vectors

Simon Hansman and Ray Hunt define the attack vector in their *A Taxonomy of Network and Computer Attacks*: "The attack vector of an attack is the main means by which the attack reaches its target."[16]

This analysis will use *The Common Attack Pattern Enumeration and Classification* (CAPEC™) to classify the attack vectors, a catalog of attack patterns with classification taxonomy. CAPEC is very useful in threat modeling. It helps to map threats to an attack surface. [17]

During this work was found that sometimes there is no relevant or detailed enough attack vector in CAPEC. In such cases is used MITRE ATT&CK taxonomy of techniques.[18]

1.5 Testing Methodology

White box testing requires deep knowledge of internal system architecture and mainly requires the availability of the source code. On the other hand, black-box testing works with no knowledge of internal architecture. Grey box testing is a middle ground, which provides combined benefits of white box and black box testing techniques. In our case, additional insight into tested products is not knowledge of source code but partial insight into the architecture of the solution and knowledge of real-world deployments of the system.

Grey box testing may have the following advantages:

- utilizes more profound knowledge of the platform
- the test is done from the user's point of view
- addresses possible internal threats
- may lead to faster results and or higher granularity comparing to black box testing

This technique also has its disadvantage concerning cybersecurity: not all possible program paths are tested.[19]

2 WHAT IS DOCUMENT MANAGEMENT SYSTEM?

Document Management System (DMS) is software that controls and organizes documents throughout an organization.[20] It adds functionality, which helps knowledge-workers effectively and securely work with documents:

- Versioning
- Metadata – primarily business metadata
- Search – full text and metadata
- Security and access control
- Storage Management

Enterprise Content Management (ECM) is a superset term covering Document Management, Records Management, Workflow Management, Web Content Management, Digital Asset Management, and others. [21]

Document Management System and Enterprise Content Management are solely understood as software solutions for electronic and digitalized paper documents in the context of this document. At the same time, these terms can also be used for strategies and methods for handling documents within organizations.

Content Services Platform supersedes Document Management Systems and Enterprise Content Management in the market. Content Services Platform is focused on integrations with line-of-business applications and coexists as a foundation platform, as per the market definition.

2.1 Typical Configuration of the Content Server Platform

The OpenText Content Server platform is an extensive ecosystem of different components and integrations. Each of these components and modules not only adds its functionality to the whole system. In addition, it sometimes enables a new functionality thanks to symbiosis with an already existing one. Adding a new component of the system not only adds functionality. It can also limit the use of some functionality of other parts. Analyzing such a platform is a complex task, which significantly exceeds the resources available for this work.

Each customer has unique requirements and integrates the Content Server with different leading applications. Fortunately, there are commonly used components. The chosen components, RDMS, Operating System, Application Server, Web Server, are used by most customers. This decision is based on experience with customers' systems. The resulting typical configuration used in this thesis reflects the current state of the years 2020 and 2021.

Customers often delay upgrades due to the cost and risks involved. The long period between the upgrades makes the single upgrade more complex. Furthermore, this added complexity leads to postponing the upgrades to an even more significant extent. Upgrading over several major versions leads to upgrades of a database, operating system, incompatible component versions. Sometimes, the only end of the installed version's support period is the only driver for an upgrade.

This typical configuration reflects this behavior. OpenText components versions are used accordingly. A typical configuration was set as a supported combination following the OpenText Product Compatibility Matrix document.

2.1.1 Typical Configuration

The typical configuration used in this thesis is:

- Content Server 16.2.5 – in a native cluster (multiple frontends and one backend)
- OpenText Directory Services (OTDS) 16.2
- Archive Center 16.2.x
- System Center 21.1.1.146
- Database – Microsoft SQL Server 2016
- Operating system – Microsoft Windows Server 2016 Standard

The latest version of the System Center is an exception in this typical configuration. System Center itself can be upgraded without affecting running applications. Only basic tests need to be done after System Center upgrade. These basic tests are limited to only the functionality of the System Center. There is no need for regression tests of other applications.

Content Server functionality can be extended by using so-called modules. Some of the modules became part of the standard Content Server distribution (e.g., WebReports). The typical configuration uses only modules, which are part of the standard distribution for the specified version of the Content Server. Security Clearance and Supplemental Markings module is not included.

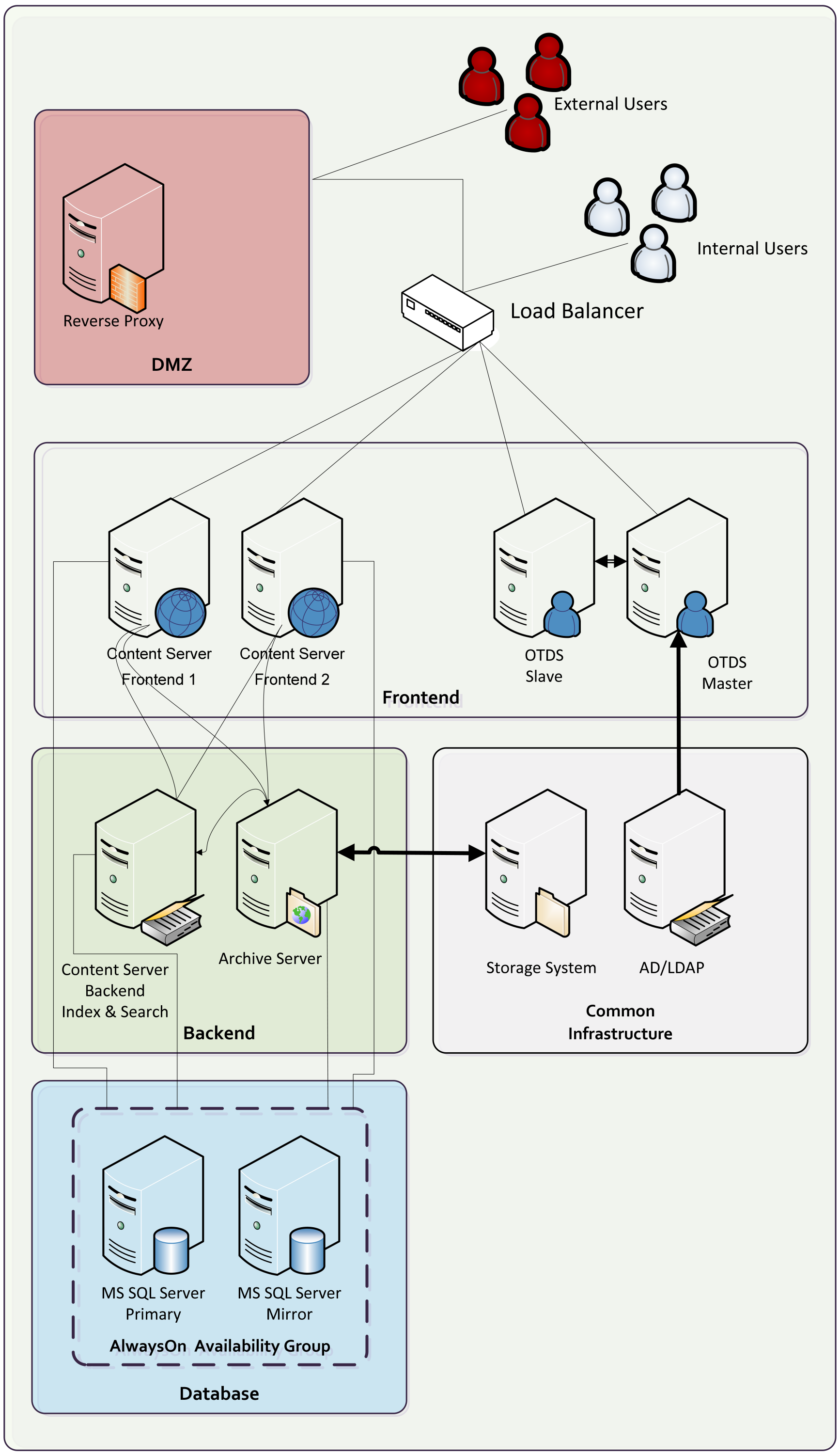


Figure 2.1 Architecture Schema [source: the author]

2.1.2 Content Server

The Content Server provides quite a broad set of functionalities. To give the reader a brief description needed to understand the Content Server functionality, we will go through a part of it in this chapter. All functionality is seamlessly integrated into the user interface.

Standard functionality:

- Versioning – there are two types of versioning available:
  - Linear versioning (integer type version names)
  - Advanced versioning (major and minor version) – this type of versioning allows different access permissions for major versions (official or published versions) and minor (working versions)
- Metadata – there are two basic types of business metadata with multi-lingual capabilities:
  - Categories – sets of attributes
  - Classification – a taxonomy tree-like the structure of text metadata
- Access control – permissions are assigned to object in the Content Server; all actions may be audited
- Storage management – flexible way of configuration, which document will be stored where.
- Document-centric workflows
  - Structured and ad-hoc routing
  - BPMN 2.0 process notation
  - Automatic escalations
- Search – full-text search, metadata search, predefined and custom search masks, possibility to save search results or run an action
- Navigation
  - Folder-like structure
  - Virtual (search) folders

  - Role-based views
  - Faceted browse (content filters)
  - Document Thumbnails – an image of the first-page document page

- Social and Collaborative functionality
  - Activity feeds
  - In-line commenting
  - Discussions
  - News
  - Task lists
  - Polls
- Forms – possibility to create custom forms and save their content into a database or create a document from form content, start a workflow with a form.
- Reports – there are two basic types of reports:
  - LiveReports – allow retrieving any information from a database
  - WebReports – offer advanced functionality to LiveReports, automation, formating, and a broad selection of different outputs

The Content Server provides two versions of web-based user interfaces:

- Classic UI – legacy user interface
- Smart UI – sometimes called Smart View, modern-looking responsive UI with better support of mobile devices, a variety of pre-configured widgets, better for integration to other business applications

OpenText provides several thick clients or extensions into thick applications. They are, however, not the subject of analysis in this thesis.

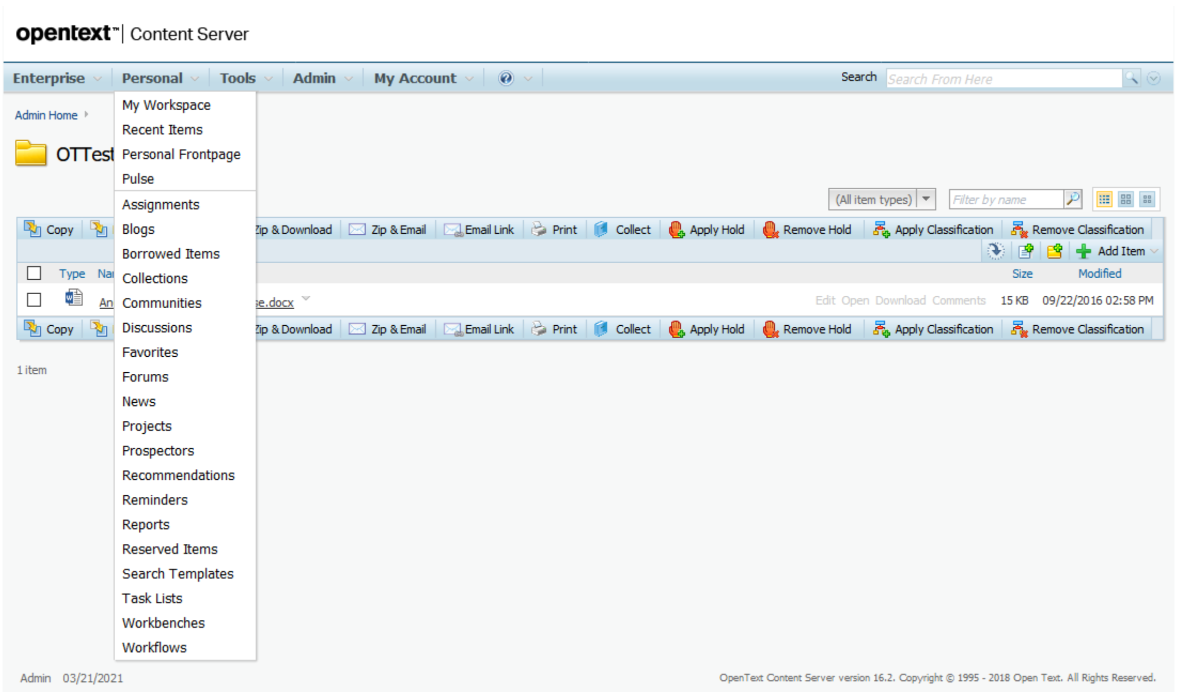
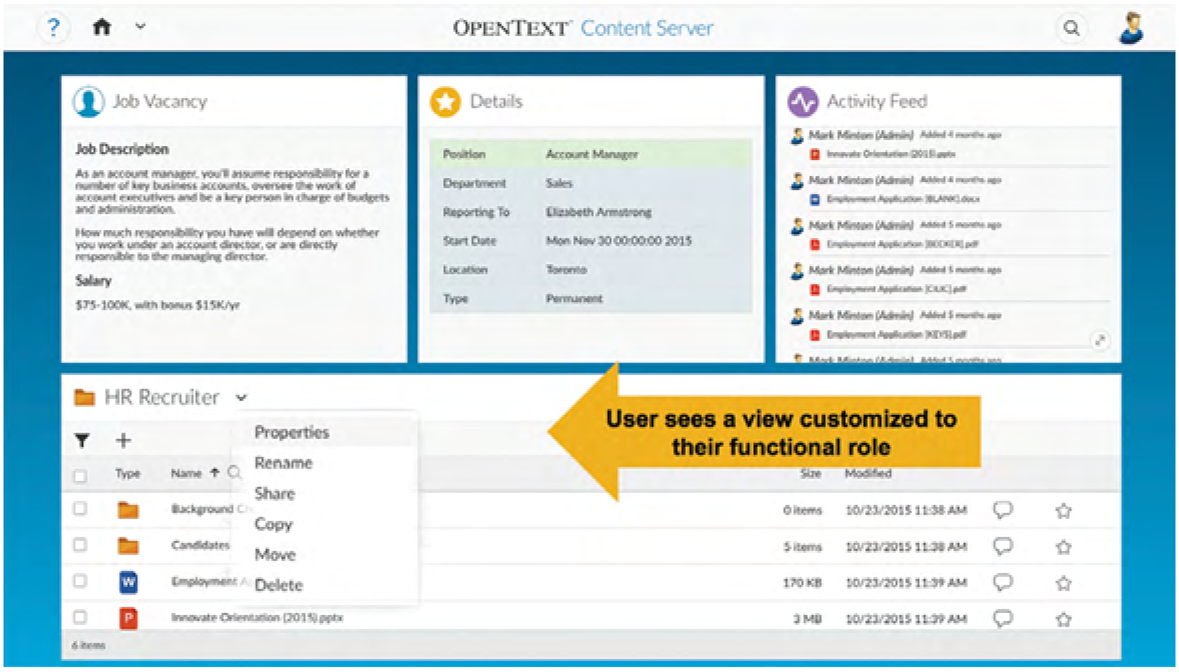


Figure 2.2 Content Server Classic User Interface [source: the author]



Job Description

As an account manager, you'll assume responsibility for a number of key business accounts, oversee the work of account executives and be a key person in charge of budgets and administration.

How much responsibility you have will depend on whether you work under an account director, or are directly responsible to the managing director.

Salary

$75-100K, with bonus $15K/yr

Position	Account Manager
Department	Sales
Reporting To	Elizabeth Armstrong
Start Date	Mon Nov 30 00:00:00 2015
Location	Toronto
Type	Permanent

Type	Name	Size	Modified
	Background C	0 items	10/23/2015 11:38 AM
	Candidates	5 items	10/23/2015 11:38 AM
	Employment A	170 KB	10/23/2015 11:39 AM
	Innovate Orientation (2015).pptx	3 MB	10/23/2015 11:39 AM

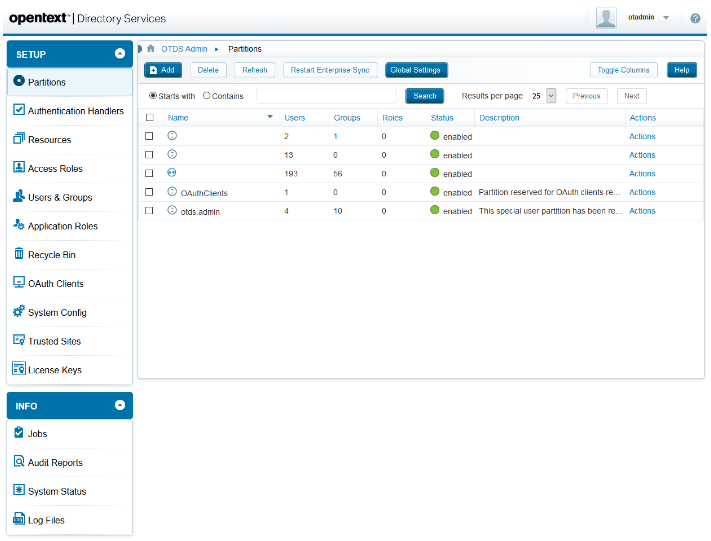
6 items

Figure 2.3 Content Server Smart User Interface[22]

2.1.3 OpenText Directory Services

OpenText Directory Services (OTDS) were built as a universal Identity and Access Management (IAM) solution for OpenText products and platforms. It manages user and group identities and provides their synchronization with leading systems, Open-Text software components, and Active Directory or LDAP-based system. In addition, OTDS provides Single-Sign-On (SSO) authentication across different systems. The main advantage is that users can seamlessly switch among different systems (for example, SAP, Oracle, Salesforce), which use different authentication methods and different usernames or unique IDs for the same user. Additionally, it is utilized for the creation of applications of licenses for OpenText products.

Partitions are self-contained databases storing information about users, groups, and organizational units. They may be managed outside OTDS and only synchronized to OTDS (called *synchronized partitions*) or fully managed within OTDS. The synchronized partitions serve as identity providers. They may represent, for example, Microsoft Active Directory, LDAP system, Oracle Directory Server. Furthermore, Authentication Handlers provide integration to Authentication Providers. OTDS has pluggable authentication architecture. It supports multiple Authentication Handlers. Only the most common ways of authentication are mentioned here: username and password, standard Web authentication protocols such as SAML2, OAuth, OpenID, supports for authentication of Windows desktop, Kerberos, Web Server (IWA), SAPSSOEXT (SAP Logon Ticket), Oracle EBS and using third-party Web Access Management products. 2-Factor Authentication (2FA) is also supported. A resource represents a system or component using OTDS as IAM. A system represented by a resource may have its own user database. The Content Server is configured as a resource in the Content Server platform. Access Roles are defined to control which users have access to which resources. Multiple OTDS servers may be deployed and used concurrently. The only synchronization with identity providers can be done from the master OTDS server.



Name	Users	Groups	Roles	Status	Description	Actions
	2	1	0	enabled		Actions
	13	0	0	enabled		Actions
	193	56	0	enabled		Actions
OAuthClients	1	0	0	enabled	Partition reserved for OAuth clients re...	Actions
otds.admin	4	10	0	enabled	This special user partition has been re...	Actions

Figure 2.4 OpenText Directory Services Administration [source: the author]

2.1.4 Archive Center

The Archive Center is also known as the Archive Server. The original Archive Server is responsible for storing "the content" of documents (files) in a scalable and secure repository. It creates a transparent layer for the management of content repositories. We can call this functionality storage virtualization.

The Archive Server's functionality was extended with basic metadata and search functionality. Furthermore, it was rebranded to the Archive Center. However, this new functionality is not utilized within the Content Server platform. Therefore, the terms Archive Center and Archive Server are used interchangeably in this thesis.

The primary use of the Archive Server is to offload the management of documents (understand files) from leading applications. It started as a solution for SAP, where documents attached to records in SAP were stored directly in a database, which slows down the database and is costly as the database needs fast disk drives and more RAM. Simply put, a document from SAP (or any other leading system) is stored in an Archive Server. The Archive Server provides a leading system with a unique ID of the stored document stored in a leading system. When a leading system requires a document back, it sends a request with its unique ID to the Archive Server. The protocol created for this purpose is called ArchiveLink and was developed together with Archive Server. Content Server platform uses Archive Server this way.

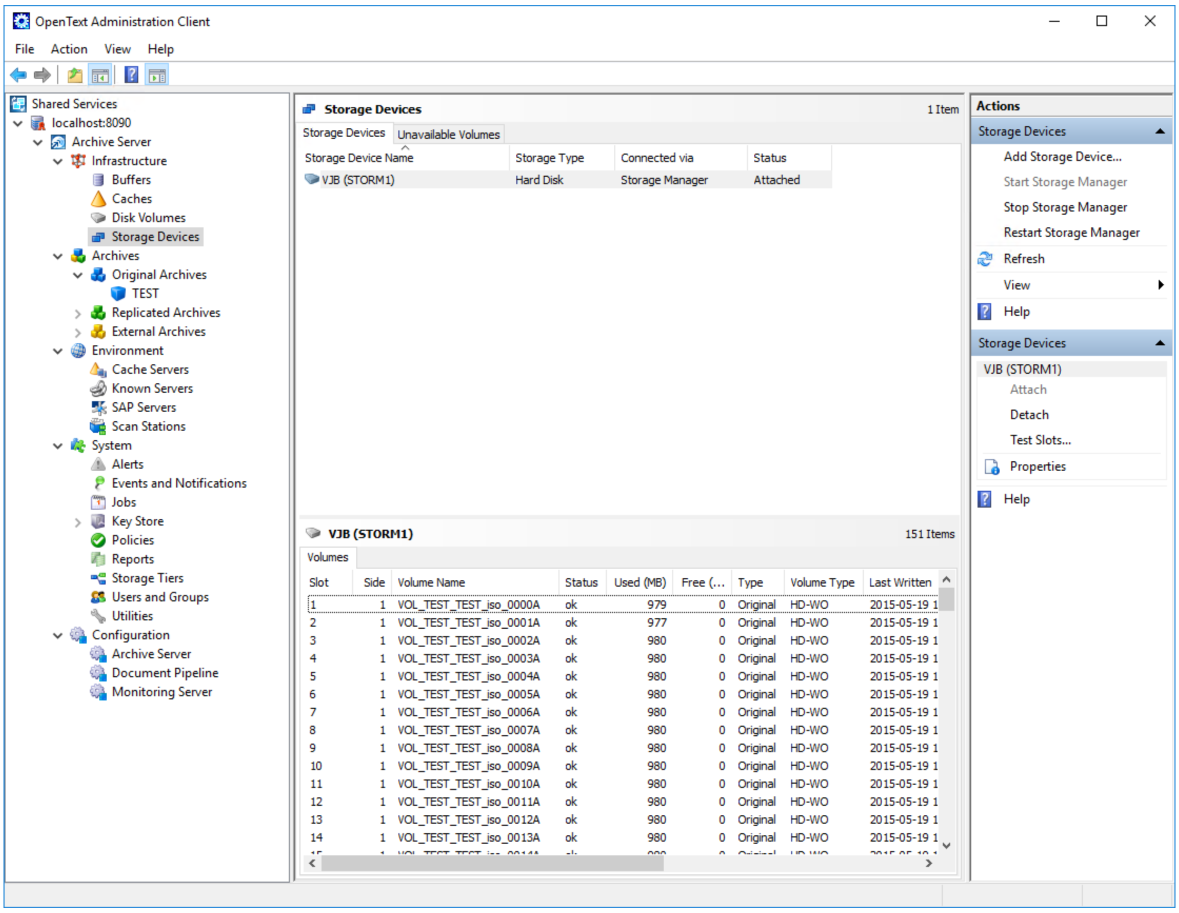
Archive Server can be configured to store documents using different storage platforms. It may vary from local or remote NAS, CAS, SAN to cloud solutions like Microsoft Azure, Amazon Simple Storage Service. An administrator may migrate documents between those solutions without impacting availability for an end-user, which is crucial functionality for the long-term preservation and readability of documents.

Archive Server handles retention periods. Documents not past their retention period cannot be deleted from the Archive Server.

Using Single file archiving for scenarios like email archiving may significantly reduce the amount of stored data. For example, instead of storing an email attachment multiple times, the attachment is stored once while it was sent to a large group of receivers. It is a similar scenario to Content-addressable Storage (CAS). Using ISO images instead of single files may be beneficial when vast amounts of small documents are archived. It will reduce the amount of space wasted bind with the cluster size of a storage system. Also, it overcomes the limitations of file systems and allows faster backups.

The concept of Known servers allows the deployment of multiple Archive Servers sharing and synchronizing their repositories for additional disaster recovery and optimization. In addition, the Archive Server supports clustering for high availability.

When users are in a location with limited bandwidth to the site with Archive Server, Cache Server may be deployed to this location to speed up reading and storing documents.



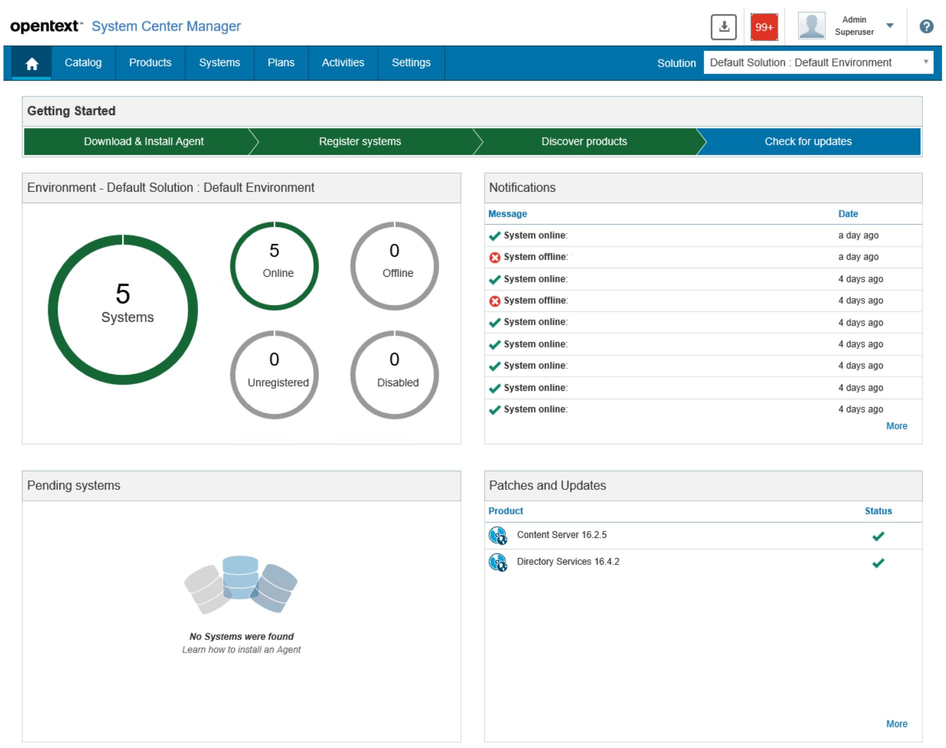
Storage Device Name	Storage Type	Connected via	Status
VJB (STORM1)	Hard Disk	Storage Manager	Attached

Slot	Side	Volume Name	Status	Used (MB)	Free (...	Type	Volume Type	Last Written
1	1	VOL_TEST_TEST_iso_0000A	ok	979	0	Original	HD-WO	2015-05-19 1
2	1	VOL_TEST_TEST_iso_0001A	ok	977	0	Original	HD-WO	2015-05-19 1
3	1	VOL_TEST_TEST_iso_0002A	ok	980	0	Original	HD-WO	2015-05-19 1
4	1	VOL_TEST_TEST_iso_0003A	ok	980	0	Original	HD-WO	2015-05-19 1
5	1	VOL_TEST_TEST_iso_0004A	ok	980	0	Original	HD-WO	2015-05-19 1
6	1	VOL_TEST_TEST_iso_0005A	ok	980	0	Original	HD-WO	2015-05-19 1
7	1	VOL_TEST_TEST_iso_0006A	ok	980	0	Original	HD-WO	2015-05-19 1
8	1	VOL_TEST_TEST_iso_0007A	ok	980	0	Original	HD-WO	2015-05-19 1
9	1	VOL_TEST_TEST_iso_0008A	ok	980	0	Original	HD-WO	2015-05-19 1
10	1	VOL_TEST_TEST_iso_0009A	ok	980	0	Original	HD-WO	2015-05-19 1
11	1	VOL_TEST_TEST_iso_0010A	ok	980	0	Original	HD-WO	2015-05-19 1
12	1	VOL_TEST_TEST_iso_0011A	ok	980	0	Original	HD-WO	2015-05-19 1
13	1	VOL_TEST_TEST_iso_0012A	ok	980	0	Original	HD-WO	2015-05-19 1
14	1	VOL_TEST_TEST_iso_0013A	ok	980	0	Original	HD-WO	2015-05-19 1

Figure 2.5 OpenText Archive Server Administration Client [source: the author]

2.1.5 System Center

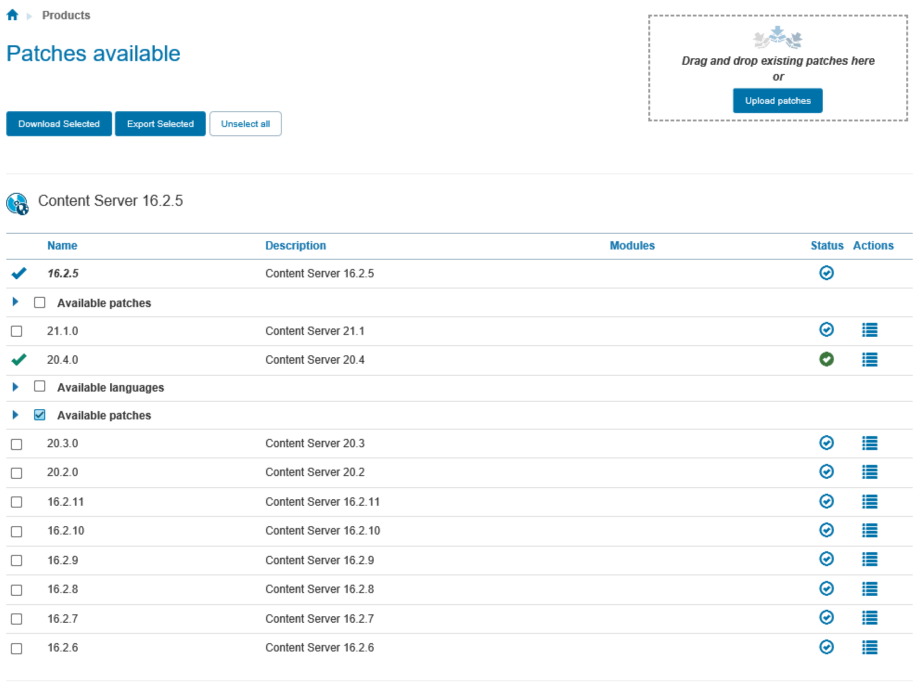
The System Center simplifies installation and updates of supported OpenText components (not all OpenText components are supported now). A System Center user triggers updates and installations. System Center offers available updates, patches. An administrator decides which update or patch to install. If System Center works in online mode (Internet access to OpenText Knowledge Base), it will download needed patches or updates, check prerequisites and install them. This part took hours or sometimes the whole day before System Center use.



Message	Date
✓ System online:	a day ago
✗ System offline:	a day ago
✓ System online:	4 days ago
✗ System offline:	4 days ago
✓ System online:	4 days ago
✓ System online:	4 days ago
✓ System online:	4 days ago
✓ System online:	4 days ago
✓ System online:	4 days ago

Product	Status
Content Server 16.2.5	✓
Directory Services 16.4.2	✓

Figure 2.6 OpenText System Center – Dashboard [source: the author]



Products

Patches available

Content Server 16.2.5

	Name	Description	Modules	Status	Actions
✓	*16.2.5*	Content Server 16.2.5		✓	
▸ ☐	**Available patches**				
☐	21.1.0	Content Server 21.1		✓	☰
✓	20.4.0	Content Server 20.4		✓	☰
▸ ☐	**Available languages**				
▸ ☑	**Available patches**				
☐	20.3.0	Content Server 20.3		✓	☰
☐	20.2.0	Content Server 20.2		✓	☰
☐	16.2.11	Content Server 16.2.11		✓	☰
☐	16.2.10	Content Server 16.2.10		✓	☰
☐	16.2.9	Content Server 16.2.9		✓	☰
☐	16.2.8	Content Server 16.2.8		✓	☰
☐	16.2.7	Content Server 16.2.7		✓	☰
☐	16.2.6	Content Server 16.2.6		✓	☰

Figure 2.7 OpenText System Center – Patches [source: the author]

II. ANALYSIS

3 ATTACK SURFACE

The following sub-chapters will describe each platform component's technical details and typical attack vectors when convenient. This background is essential for describing the attack surface of the platform. This thesis is an initial work attempting to map possible ways to attack the Content Server platform. Enumerating and evaluating all possible weaknesses of the platform is a too complex task. However, the reader can understand it as groundwork, which can be used to build on.

The typical configuration of the Content Server platform consists of the following components:

- Content Server
- OpenText Directory Services
- Archive Center
- System Center

The OpenText Content Server platform uses different third-party party components in its implementation. As the vast majority of these components are well known, their vulnerabilities are known as well. In addition, some of them have available exploits. Therefore, an essential part of this chapter is to map used third-party components, determine their version used in the implementation of the platform, check their known vulnerabilities, and determine if these vulnerabilities are a relevant threat to the platform. These steps will be done for a limited set of third-party components due to the complexity of this task. The components chosen for the detailed analysis are those, which are more significant and visible. However, it does not mean those generate the highest risk for the platform.

Important third-party components used within the Content Server platform:

- OpenDJ – OTDS
- Tomcat – OTDS, Archive Center
- IIS – Content Server
- Eclipse Jetty – System Center

Users or administrators of all components can use OTDS authentication. However, the Archive Center and the System Center can use their local authentication functionality.

The Content Server and the Archive Center use external RDBMS. OTDS and the System Center use embedded solutions for storing their data.

The Archive Center uses file systems as temporary storage or permanent storage of the content. The Content Server uses file systems to store full-text indexes.

The whole platform has been on the market for a long time. Many consultants are setting up customer systems for years, and each customer has several environments to follow ITIL rules. Moreover, the vast majority of them are non-production systems. This circumstance creates situations where passwords may have a default value or simple common passwords are reused. Thus, the systems may be prone to attack vectors like:

- CAPEC-70: Try Common or Default Usernames and Passwords[23]
- CAPEC-565: Password Spraying[24]

Alternatively, an attacker may use a standard attack vector for web applications and try Steal Web Session Cookie.[18]

Additionally, as these systems are heavily integrated, an attacker may gain access via other integrated systems or leading applications.

3.1 Content Server

The Content Server offers two versions of a browser-based user interface. The older one is called Classic UI, and the newer called Smart UI. Smart UI is built using HTML5 and Node.js.

Just to mention it here, Content Server with web UI may be approached with standard attack vectors like *Man in the Middle Attack* (CAPEC-94) [25] or *Denial-of-Service* attack tailored for the Content Server *Sustained Client Engagement* (CAPEC-227). [26]

The simplest native Content Server clusters consist of multiple frontend servers and one backend server, where:

- Frontend server(s) – the communication layer for clients
- Backend server – the application layer processes all of the requests, holds the logic for all transactions, and provides search and index services responsible for full-text search and indexing.

Content Server consists of two services:

- OTCS – service provides web user interface, in our case using IIS as a web server.
- OTCSAdmin – backend service, provides search and index functionality, running the vast majority of agents

The Content Server uses JDBC to connect the database. Database connection info is stored in the dbconnection section of opentext.ini configuration file. The password of the database user is included as an encrypted string. The same plaintext password may have a different encrypted string on different machines. It stays the same only for cloned environments. This behavior prevents the use of the attack vector *Rainbow Table Password Cracking* (CAPEC-55).[27]

Authentication of the user is done using OTDS. When the user accesses a Content Server URL and does not have a valid authentication cookie, the connection is redirected to OTDS. After successful authentication, the cookie is stored. Then, the connection is redirected back to the original requested Contest Server URL.

Audit information is stored in two database tables. Theoretically, it may be possible for an attacker to clean up the audit log by deleting specific records or truncate audit tables. As far as this work analyzed the database schema, there is no mechanism to check if the Audit log was modified outside the Content Server. So, it may be prone to attack vector CAPEC-268: *Audit Log Manipulation*[28] Of course, an attacker with access to a database does not have to use UI or API to get the needed information. However, this scenario would require knowledge of the Content Server database schema. It is more complicated than deleting specific records from two tables. Moreover, it is possible to modify the database table, which instructs the Content Server which events to add to the audit log. Installation of the Security Clearance and Supplemental Markings module should prevent such a modification by encrypting database records. Furthermore, administrators have available legitimate functionality for purging the audit log as it can grow fast.

The Content Server serves as central document storage in an organization and may be used to disseminate infected MS Office files or malicious links. Users will suspect less file, image, or link coming from an internal system than something coming via email (all sub-techniques for *User Execution* T1204. [29]) Another level will be the use of legitimate business workflow for *Internal Spearphishing*.[30] It is possible to configure the Content Server to block the ingestion of files based on their MIME types. Currently, the primary attack vector is malicious MS Office files. Preventing ingestion of these files will go against the vital functionality of DMS. Content Server does not offer the functionality to scan files during upload for malware. It is expected that all endpoints are adequately protected and monitored for malware. The only known way to scan uploaded documents is to use the OpenText partner's PVA Virus Scan Integration Module.[31]

The Content Server contains OScript Virtual Machine (OVM) and Java Virtual Machine (JVM) for interpreting code. OScript is the object-oriented fourth-generation programming language. The Content Server implementation is based on OScript. In addition, Java code is heavily used for the implementation of Search & Index functionality.

Patches for the Content Server are distributed as zip files. Each patch contains a folder named patch containing the main patch file. It is plaintext file named pat<patch number>.txt. It contains basic information about the patch and possible OScript code. If there are any other files, they are put into a subfolder following in exact structure of the Content Server installation folder. So, manual patching may be done by a copy of patch content to the root folder of the Content Server installation folder. When there is a newer version of any file, the old file is overwritten. During the start of the Content Server, all patch files in the patch folder are read and executed in the order defined by patch number. It may be possible for an attacker to create a malicious patch file if she can copy it remotely to the patch folder. Next time the Content Server is restarted, malicious code will be executed. This procedure is a nice possibility of getting persistence on the machine. Furthermore, it will be hidden from standard anti-malware procedures. CWSS score for this weakness is 23.9 with CWSS vector:`TI:H,0.9/AP:P,0.9/AL:A,1/IC:N,1/FC:T,1/RP:RU,0.7/RL:A,1/AV:L,0.5/AS:N,1/IN:A,1/SC:A,1/BI:M,0.6/DI:L,0.2/EX:H,1/EC:UK,0.5/P:UK,0.5/`

Content Server frontend sends requests for searching to the Admin Server located on backend Content Server using port 5858 (default value). Each component of Search & Index has its port set during the creation of Search & Index Data Source. Each Data Source has its own set of these components with their unique port numbers. The number of used ports depends on the configuration and size of the search index. The OpenText proprietary protocol is used for communication between those components. This architecture allows having several backend servers to share the load of indexing and search when needed.

3.1.1 Publicly Known Vulnerabilities

There is a list of 38 vulnerabilities on [32]. All but one are for different OpenText products or outdated Content Server versions.

The relevant vulnerability is CVE-2021-3010. [33] Its CVSS score is 5.4, and the vector is CVSS: 3.1/AV:N/AC:L/PR:L/UI:R/S:C/C:L/I:L/A:N This vulnerability is discovered in version 20.3, and also simple exploit was created. The exploit describes two possible ways for Stored Cross-Site Scripting. One is the possibility to add JavaScript code to a document version description. The second is the possibility to create a banner loading JavaScript for the Content Server Project. When the user accesses the Project page, JavaScript will be executed.[34]

OpenText sanitizes HTML and text input with AntiSamy.[35] However, there are many different Content Server modules offered by OpenText, and also customers can create their own. Combinations of modules offer much additional functionality. It would be interesting to analyze if these combinations may introduce some exploitable weaknesses in the future.

3.1.2 Additional Third-party Components

The third-party component list for the Content Server would benefit any research or map-ping of the OpenText attack risks. However, this is not included in this thesis because there is seemingly — based on the attempted search — a lack of comprehensive and complex sum-mary on this topic. Hence, the summary of additional third-party components is left open for further research, and the attempt of a thorough summary will not be taken up in this work.

3.2 OpenText Directory Services

OTDS uses OpenDJ as an embedded Directory Server because it enables synchronization, replication, high performance, and high availability.[36] User Interface for management and authorization is implemented as an Apache Tomcat web application. OTDS is LDAPv3 compliant. It offers REST API.

Installation of OTDS will create partition otds.admin. It is a non-synchronized administrative partition with predefined default groups. The administrator user *otadminotds.admin* is added as a member to all these groups. This administrator user has all privileges in OTDS.

3.2.1 Apache Tomcat

OTDS 16.2 supports Apache Tomcat 8.5 and Tomcat 9.0M. To simplify the setup of the Content Server platform based on the typical configuration, the same version of Apache Tomcat is used with Archive Server. Rarely are used different Apache Tomcat versions for Archive Server and OTDS in real-world scenarios.

Following weaknesses and attack vectors were identified for Apache Tomcat 8.5.35 in the context of this work:

- Default configuration exposes information about the exact Apache Tomcat version, which is defined as CWE-497: *Exposure of Sensitive System Information* to an *Unauthorized Control Sphere.*[37] It may be used for *Web Application Fingerprinting* (CAPEC-170).[38] Mitigation of this weakness is described further in chapters 4.3 and 5.1.
- Without further configuration changes, the default configuration of Apache Tomcat uses outdated secure transport protocols. It is the weakness classified as CWE-326: *Inadequate Encryption Strength*[39]
- Apache Tomcat has configured the shutdown port 8005 by default. This functionality allows anyone connecting to port 8005 to shut down Apache Tomcat without any authentication. This functionality may be abuse for a denial-of-service attack. CAPEC classifies this as CAPEC-227: *Sustained Client Engagement.*[26] An Apache Tomcat configuration may be modified to allow connection only from a specified address. Furthermore, it allows configuring custom string for the shutdown command. However, the safest way is to disable the shutdown port.
- The default configuration of OTDS and Apache Tomcat does not restrict manipulation with cookies using JavaScript. It may allow an attacker to steal OTDS authentication cookie using Cross-Site-Scripting and impersonate as properly au-

thenticated user by OTDS. This weakness is classified as CWE-1004: *Sensitive Cookie Without 'HttpOnly' Flag*[40]

There are 45 publicly known vulnerabilities of Apache Tomcat 8.5.x and 52 vulnerabilities of version 9. For the complete list, refer to Appendix A I.

Following vulnerabilities were analyzed:

- CVE-2016-8735 Apache Tomcat JmxRemoteLifecycleListener access control

  The default configuration does not enable the listener, and it is not required for OTDS and Archive Server functionality.

- CVE-2016-3427 Apache Tomcat JmxRemoteLifecycleListener privileges management

  The default configuration does not enable the listener, and it is not needed for OTDS and Archive Server functionality.

- CVE-2017-5648 Apache Tomcat Application Listener access control

  This vulnerability is applicable only if Security Manager is configured. Using Security Manager would improve the overall security of Apache Tomcat and should be investigated in the future.

- CVE-2020-1938 Apache Tomcat AJP Connector Ghostcat input validation

  Base Score: 9.1 CRITICAL

  CVSS Vector: `3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N`

  Mitigation of this vulnerability is described further in the chapter 5.3.

3.2.2 OpenDJ

The analyzed version of OTDS (16.4.2) includes OpenDJ Community Edition 2.6.4. OpenDJ is implemented in Java.

The default configuration of OTDS is using the following ports:

- 8989 – Secure replication port
- 389 – LDAP Connection handler port
- 636 – LDAPS Connection handler port (disabled)
- 1689 – JMX Connection handler port (disabled)

Acquiring remote access to the OTDS machine for an attacker enables her or him to get complete control of the Content Server platform authentication. There is no need to get the password of any existing OTDS user. The OpenDJ superuser *Directory Manager* password can be changed by modifying SHA512 encoded password string within configuration file `%OTDS_HOME%\opendj\config\config.ldif`. The password of this user can also be changed using the command-line tool. OpenDJ is such a critical component that this should be classified as a weakness *Weak Password Recovery Mechanism for Forgotten Password* (CWE-640). [41] In case the password is forgotten there should not be possible to recover password without losing data in the Directory Server.

Password itself is encoded using Salted SHA-512 algorithm. Salt is 8-bit long. Encoded string is stored in the following format: `"{SSHA512}" base64(<digest> <salt>)`[42]

It will be helpful to investigate in the future if an attacker can create a user directly in OpenDJ and hiding it from the OTDS interface. Furthermore, acquiring a hidden persistent presence in the system.

There are 42 publicly known issues in the OpenDJ Release Notes, and additional issues found after release are available in Security Advisories. [43] In addition, the following issues were investigated as part of this work:

- Issue #201703-02: Sending random data to LDAP/LDAPS ports may expose information about the service [44]

When nonsensical data are sent to LDAP or LDAPS ports, and OpenDJ cannot decode it, OpenDJ responds with information about the application's expected protocol and qualified Java class name. After this, a connection ends. This weakness is classified as CWE-497: *Exposure of Sensitive System Information to an Unauthorized Control Sphere.*[37] It may be used for *Application Fingerprinting* (CAPEC-541) [45] Mitigation of this weakness is described further in the chapter 5.7

- Issue #201706-02: SASL security layer may use excessive memory[46]

  Buffer size is not limited to data handover between client and server when using the DIGEST-MD5 and GSS-API SASL mechanisms. This weakness allows using an excessive amount of memory. An attack using this weakness is classified as CAPEC-603: *Blockage.*[47] Mitigation is described in the chapter 5.9

- CVE-2014-3566 POODLE SSL Vulnerability[48]

  OpenDJ uses SSL version 3.0, which is prone to POODLE SSL vulnerability.[49] It is classified as *Inadequate Encryption Strength* (CWE-326) weakness. [39] Currently, not only SSL 3.0 but also TLS version 1.0 and 1.1 are not considered safe enough. The solution to this weakness is described further in the chapter 5.8.

3.2.3 Additional Third-party Components

OpenText provides the list of third-party components used for OTDS implementation as a file otds.installation-16.2.0-rod3rdPartyLicenseTexts.html.[50] The list within this file contains 68 unique versions of third-party components. Therefore, it may be helpful to check how these components influence the security of OTDS in the future.

3.3 Archive Center

The Archive Server consists of three core components: Administration server (ADMS), Document Service (DS), and Storage manager (STORM).

- The Administration Server's purpose is to create and maintain the environment of the Archive Server. It also serves as the interface to the Administration Client and other tools.
- Document Service is responsible for storing and retrieving documents and their components. Each document consists of single or multiple components.
- Storage Manager manages and controls the storage devices.

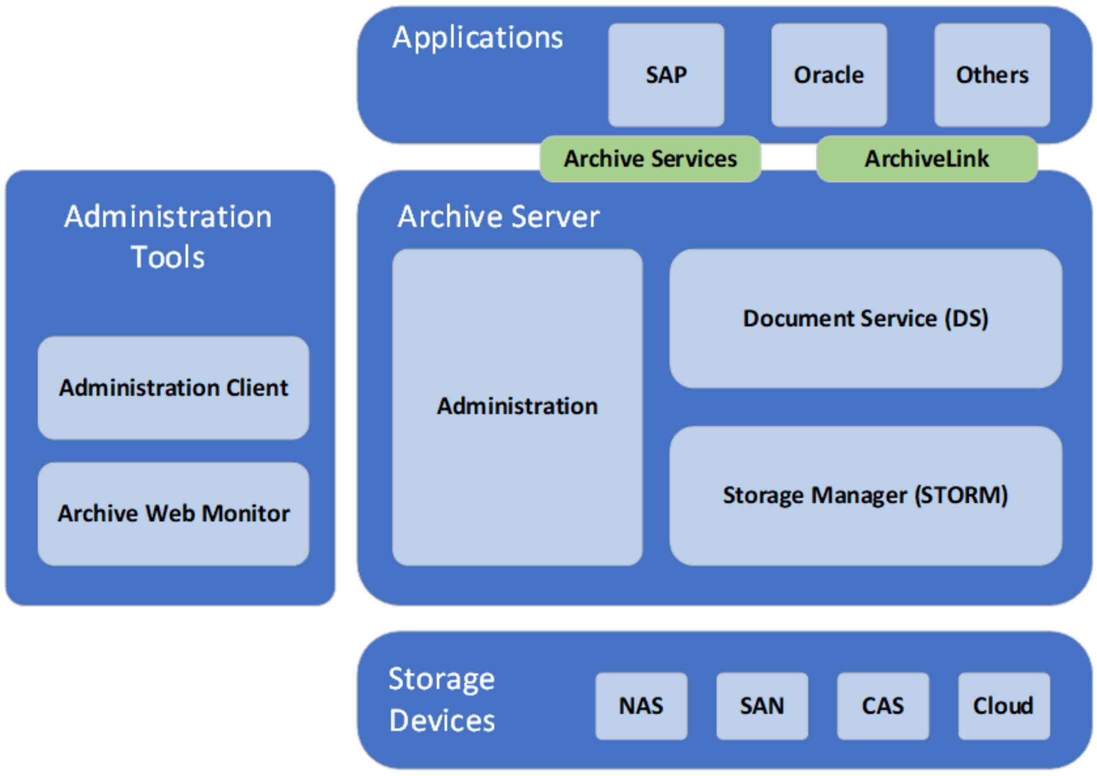


Figure 3.1 Archive Server Architecture Schema [source:the author]

The Archive Monitoring Server is another Archive Center component responsible for collecting statistics about archiving and retrieving activities and performance of the Archive Server. It monitors the error queues of the Archive server components and reports quota limit violations.[51]

Administration Client is a GUI application to manage settings of Archive Server, a thick client based on Microsoft Management Console 3.0. Administration Client connects to Archive server using HTTP or HTTPS protocols.

Archive Monitoring Web Client is a browser-based application used to monitor information regarding the status of relevant processes, the file system, the database size, and available resources.

Document Pipeline and Document Pipeline Info client would not be described in this document as they are not utilized in our typical configuration of the Content Server platform.

Archive Server uses RDBMS to store technical info about documents and some parts of its configuration. Database user credentials are stored as encrypted strings in a plaintext configuration file, together with and additional connection info. However, encryption provides the same pair of plaintext and encrypted (or hashed) string on all installations of the Archive Server. This weakness is classified as *Weak Encoding for Password* (CWE-261). [52] It can be abused using the technique *Rainbow Table Password Cracking* (CAPEC-55). [27] A rainbow table can be created using the command line utility dsClient and its enc command.

Technically, the Archive Server consists of two services:

- Apache Tomcat running code, which is written in Java
- Spawner service is written in C++

Spawner is responsible for starting other Archive Server processes. It starts processes defined in plaintext servtab files of the servtab subfolder of the configuration folder. For an attacker with remote access to the Archive Server machine, modifying or adding a new servtab file instructs the Spawner service to run provided binary. Thus, it is a possibility to get persistence on the machine. Moreover, it would not be so apparent as to modify Windows service directly. The CWSS score for this weakness is 18.1 with CWSS vector: `TI:M,0.6/AP:P,0.9/AL:A,1/IC:N,1/FC:T,1/RP:RU,0.7/RL:A,1/AV:L,0.5/AS:N,1/IN:A,1/SC:A,1/BI:M,0.6/DI:L,0.2/EX:M,0.6/EC:UK,0.5/P:UK,0.5/` An attacker also can use the attack vector *Modification of Windows Service Configuration* (CAPEC-478)[53] for gaining *SYSTEM* privileges.

Archive Server may use a built-in User Management system or OTDS for user authentication. Users and groups within built-in User Management are intended to administer the Archive Server and its associated components like Document Pipeline. Standard scenarios do not use them for the retrieval and storage of documents. The administration user having all privileges in the Archive Server is called dsadmin. This user has no password after installation. A quite often, it stays unchanged. This work understands it as *Weak Password Requirements* (CWE-521),[54] which may be misused. Possible attack vector will be classified as CAPEC-70: *Try Common or Default Usernames and Passwords.*[23]

The Archive Monitoring Web Client is available at the following URL:`http://<server>.[<domain>]:[<port>]/archive/monitoring`. Basic authentication is used, and it is not secure without HTTPS protocol. An attacker capturing network communication can acquire the administrator's credentials to log in to the Archive Monitoring Web Client. Especially when the Archive Server is configured with OTDS authentication, these credentials may be used to access the whole Content Server platform.

The Archive Monitoring Server can be configured to gather information about a number of components and the data volume downloaded by a specific user per defined period (default is 30 days). If this quota is exceeded, an event within the Archive Server Administration Server is created, and notification or automatic action may be triggered.

Leading applications use the ArchiveLink interface to store and retrieve documents from the Archive Server. ArchiveLink is HTTP-based. An URL requesting action with a document must contain at least a Logical Archive name (Logical Archive is also known as Content Repository in SAP environments), DocID (unique ID identifying the document within the Archive Server), and ArchiveLink protocol version. Example of request for document retrieval: `http://192.168.1.61:8080/archive?get&pVersion=0045&contRep=FS&docId=aaj5ga6v4csedizmkiaafhr4aguma`

The Archive Server uses SecKeys as a way for authenticating incoming requests. First, the ArchiveLink URL from the previous example must be extended with Access Mode code (representing the following privileges: read, create, update, delete), Authorization ID (ID of the user from the leading application), and expiration timestamp. Then, message Digest (MD) is created from a string with all these attributes using the MD5 or RIPEMD-160 algorithm. Finally, it is signed using a certificate of the leading application. This public part of this certificate must be imported into the Archive Server and enabled before its use. This digest is attached at the end of the request URL. SecKeys are configured on the Logical Archive level.

When Seckeys is not enabled, it is possible to retrieve documents without any authentication. Only DocID must be known to an attacker. DocIDs are semi-randomly generated. Moreover, it may be possible for an attacker to use brute force or guess the next DocID. The default configuration of Archive Server is set to prevent this scenario. The Archive Server will log a warning after a certain amount of unsuccessful retrieval requests in a row for unknown DocIDs. However, this type of log message is quite rarely monitored.

Timestamps may be enabled for Logical Archives to ensure that documents were not modified and were present in the system at the date and time of timestamping. It is also possible to set a level of timestamp checking during retrieval of documents from not serving documents with a not valid timestamp to ignoring timestamps at all.

Archive Server has functionality allowing encryption of archived documents. Encryption prevents anybody with remote access to the Archive Server, remote access to the file system of the Archive server, or access to a remote storage system from reading the content of documents.

Description of Archive Center DMS functionality added on top of Archive Server functionality will be omitted here because it is not utilized in the Content Server platform. However, it may be worth checking if this unconfigured functionality or basic configuration cannot be misused for getting documents stored in the Archive Server.

3.3.1 Apache Tomcat

Archive Server version 16.2.0 was released with the support of Apache Tomcat version 8.5.13+ Archive Server update 16.2.3 enabled support of Apache Tomcat 9.0.x. However, according to experience, a vast majority of a real-world installation of Archive Server still uses Apache Tomcat 8.5.x. This experience led to using this older version (specifically 8.5.35) in a typical configuration of the Content Server platform.

The same version of Apache Tomcat is used for OTDS and Archive Server; the same attack vectors and weaknesses are relevant for Apache Tomcat used for OTDS. More information in the chapter 3.2.1

3.3.2 Additional Third-party Components

OpenText provides the list of third-party components. [50] This list consists of 199 different component versions (some of the components are used in several versions in the Archive Server implementation). Some of the components are specific for Operating System, RDBMS, or only for the installation process of the Archive Server itself. This list could be significantly reduced after filtering only relevant components. However, this goes beyond of timeframe available for this analysis.

3.4 System Center

System Center has two main parts. The System Center Manager, which is the server part. And the System Center Agent, the client part of System Center.

System Center gets information about available products, versions, allowed combinations, patches, superseded patches via the Manifest file. The Manifest file is an XML file containing mentioned information for all supported OpenText applications. It is automatically updated from the OpenText Knowledge Base server. If online access is not allowed, an administrator can manually upload the Manifest file, installation binaries, and patches to the System Center Manager. More information about the Manifest file is provided in the following chapter.

The System Center Manager browser-based user interface is built on Eclipse Jetty 9.4.x. The default configuration uses port 8888 for HTTP. It can be changed to a different port number. Encrypted communication (TLS) is not configured by default. The typical port number for encrypted communication is 8443.

System Center is using H2 DBMS in embedded more. Interesting will be to check the possible local connection to this DB in the future and find out if it is possible to circumvent security features of the System Center described in the following chapters.

There are two basic possibilities for user management in the System Center. It can use a local user database, or it can be connected to OTDS. The System Center is often used with built-in user management. Thus, it may be helpful to check whether there is any possibility to circumvent it in the future.

Besides installing patches to installed OpenText software components, System Center can install OpenText products and their prerequisites to machines with installed the System Center Agent.

The main reasons why the installation of patches and updates is often delayed in enterprises are risks bind with an upgrade, offline time needed for often crucial systems, and costs bind with regression and integration testing. The System Center use may only reduce needed offline time. However, there is no reason to delay updates of the System Center. The Content Server platform does not need to be taken offline while the System Center is updated. The System Center is used only by administrators, and possible failure during the upgrade does not limit end-users of the platform. Upgrade itself is just a couple of clicks in online mode. There is no need for integration testing. Moreover, a basic regression test is sufficient. The System Center may be kept on the latest version available.

Execution plans allow an administrator to run necessary actions on all machines running its agent. Thus, the most universal and powerful is the possibility to run PowerShell scripts. Unfortunately, the execution plans functionality may be abused to:

- *Indirect Command Execution* [55]
- *Lateral Tool Transfer*[56]
- *Command and Scripting Interpreter: PowerShell*[57]

Using CAPEC classification, this capability offers the attack vector for *Rogue Integration Procedures* (CAPEC-524).[58]

3.4.1 The Manifest File

The manifest file contains info about available software versions, patches, and dependencies.

The manifest file is a zip file containing a set of XML files. The settings.xml file contains a checksum of the manifest file, digest, root of the URL for download of binaries and signature, besides other information. Manifest file signature and checksum are verified during the upload of an updated manifest to the System Center. This verification prevents the use of a doctored manifest file.

Component-specific manifest files are in the catalog folder of the manifest file. These files are necessary for the automation of an installation process. Each binary referenced in these manifests contains a download URL, hash of binary, and installation prerequisites like supported Operating System, Database Server, Application Server, and other software components.

The product_manifest folder contains files holding information about each System Center supported OpenText component. Info contained in these files is necessary for an update process. In addition, there is a file for each combination of an OpenText component and an Operating System. Including System Center itself. Each contains info about available patches, their dependencies, a status if a patch is retired, download URL, and additional information about a patch itself and its installation process. Besides the info mentioned above, these files contain:

- A hash for each file from a specific version or update level – using this information System Center should check if a correct version of any installed file.
- A hash for each file deployed by a specific patch – may help determine if a patch was installed successfully.

The integrity check is run during the upload of the manifest file, patch, update, or OpenText software component. This procedure successfully prevents attacks using modified OpenText binaries (classified as CAPEC-523: *Malicious Software Implanted*[59]) or modifying the manifest file to hide or circumvent this controlling procedure. However, the upload of binaries using *external vendor file* for third-party components does not include the previously mentioned check as expected; a hash of the uploaded file is created. Thus, it prevents rewriting it with malicious code in the local file system of the System Center Manager machine.

3.4.2 Eclipse Jetty

The System Center User Interface is implemented in Eclipse Jetty. The System Center version 21.1.1.146 used in the standard configuration in this work contains Eclipse Jetty Client version 9.4.33.v20201020.

There are 21 known vulnerabilities for Eclipse Jetty.[60] Relevant vulnerabilities for this version are:

- CVE-2021-28165, CVE-2020-27223, CVE-2018-12545, which may be used for *Denial-of-Service* attack. However, as the System Center is intended only for administrators and not really for daily use, the impact will be low.
- CVE-2017-9735 (CWE-200: *Exposure of Sensitive Information to an Unauthorized Actor*[61]), CVE-2017-7658 (CWE-444: *Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling')*[62]), and CVE-2017-7657 may lead to authentication bypass, and it will be interesting to look into them in the future.
- CVE-2020-27216 – only relevant for Unix like systems
- CVE-2021-28163 (CWE-59: *Improper Link Resolution Before File Access ('Link Following')*[63]) – possible low impact vulnerability

3.4.3 Additional Third-party Components

The third-party component list for the Content Server would benefit any research or map-ping of the OpenText attack risks. However, this is not included in this thesis because there is seemingly — based on the attempted search — a lack of comprehensive and complex sum-mary on this topic. Hence, the summary of additional third-party components is left open for further research, and the attempt of a thorough summary will not be taken up in this work.

4 REDUCING ATTACK SURFACE

"Intuitively, the larger the attack surface, the more likely the system will be attacked, and hence the more insecure it is." [64]

The following chapters describe basic info about needed changes and possible settings.

4.1 Disabling Weak Transport Protocols and Ciphers

The crucial security aspect for any web-based application is to secure communication between a server and a client. However, use of weak transport protocol or cipher (CWE-326: *Inadequate Encryption Strength*[39]) and a possibility to downgrade transport protocol (CWE-757: *Selection of Less-Secure Algorithm During Negotiation*[65]) may enable an attacker to sniff network traffic (CAPEC-158: *Sniffing Network Traffic*[66]) or use *Man in the Middle Attack* (CAPEC-94)[25] and extract credential or another sensitive information. Following transport protocols are not considered secure anymore: TLSv1.1, TLSv1.0, SSL 3.0, SSL 2.0[67][68]

Weak cipher algorithms:

- RC4 cipher suites are prohibited for use with TLS from February 2015 by memo RFC 7465.[69]
- MD5 should not be used for the message integrity as it is prone to hash collision weakness.[70]
- 64-bit block cipher 3DES should be disabled due to the *Birthday attack* (SWEET32).[71]

The following sub-chapters will advise which protocols and cipher suits is possible to use.

4.1.1 IIS

Windows Server 2016 Standard IIS has following transport protocols enabled after installation of the Content Server platform: TLSv1.0, TLSv1.1, TLSv1.2

Following protocols should be disabled: TLSv1.0, TLSv1.1, SSL 2.0, SSL 3.0, PCT 1.0 and Multi-Protocol Unified Hello.

Following ciphers are offered by IIS using default configuration of the platform for TLSv1.0:

- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_RC4_128_SHA
- TLS_RSA_WITH_RC4_128_MD5

For TLSv1.1 default enabled ciphers are:

- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_RC4_128_SHA
- TLS_RSA_WITH_RC4_128_MD5

TLSv1.2 has following ciphers enabled:

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_GCM_SHA384
- TLS_RSA_WITH_AES_128_GCM_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA256
- TLS_RSA_WITH_AES_128_CBC_SHA256
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_3DES_EDE_CBC_SHA
- TLS_RSA_WITH_RC4_128_SHA
- TLS_RSA_WITH_RC4_128_MD5[72]

To enhance security of the platform we should disable following ciphers:
NULL, DES56/56, all RC2 ciphers, all RC4 ciphers

Only the following cipher suites for TLS handshake should stay enabled in this preferred order:

1. TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
2. TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
3. TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256

4. TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
5. TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
6. TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
7. TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
8. TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
9. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
10. TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
11. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
12. TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA

4.1.2 Apache Tomcat

Combination of Apache Tomcat 8.5.29 and Oracle Java 1.8.0_181 offers following transport protocols by default: TLSv1.0, TLSv1.1, TLSv1.2

Only following cipher suites for TLS handshake should stay enabled in this preferred order:

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384

Upgrade of Java to OpenJDK11 adds TLSv1.3 protocol and these additional cipher suits:

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256

- TLS_DHE_RSA_WITH_AES_128_CCM
- TLS_DHE_RSA_WITH_AES_128_CCM_8
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CCM
- TLS_DHE_RSA_WITH_AES_256_CCM_8
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_DHE_RSA_WITH_ARIA_128_GCM_SHA256
- TLS_DHE_RSA_WITH_ARIA_256_GCM_SHA384
- TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA
- TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA256
- TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA
- TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA256
- TLS_DHE_RSA_WITH_CHACHA20_POLY1305_SHA256
- TLS_ECDHE_RSA_WITH_ARIA_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_ARIA_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_CAMELLIA_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_CAMELLIA_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256

Based on findings during this analysis, recommended cipher suites for Apache Tomcat are (in this order):

1. TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
2. TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
3. TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
4. TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

4.2 HTTP Security Headers

HTTP Security Headers help improve web application security. They instruct a client (browser) how to behave. This forced behavior modification can easily prevent common attack vectors like Clickjacking, cross-site scripting, code injection, content sniffing. However, on the other hand, they can also break the functionality of a web application. As part of this work, the optimal use of HTTP Security Headers was tested for the Content Server platform.

Following HTTP security headers were evaluated:

- Strict Transport Security (HSTS)
- Content-Security-Policy
- Expect-CT (Certificate Transparency)
- Referrer-Policy
- X-Content-Type-Options
- X-Permitted-Cross-Domain-Policies
- X-XSS-Protection
- X-Frame-Option

The following sub-chapters will briefly describe their use and possible limitations in the context of this work.

During the analysis of how to properly enable security headers for the platform appeared the following question: What exactly happens when a browser gets the same security header twice? Each instance with a different protection level. Does it lower protection? The question is going to be a topic for future analysis.

4.2.1 HTTP Strict Transport Security

HSTS prevents Man-In-The-Middle (MITM) attacks using techniques like protocol downgrade, cookie hijacking. HSTS may mitigate some development and deployment bugs by blocking loading some web page resources over HTTP. This policy is defined in RFC 6797.[73]

Requirements for HSTS:

- Valid SSL certificate
- Redirection of all HTTP links to HTTPS using permanent redirect (code 301)
- HSTS header Strict-Transport-Security must be sent from the base domain for HTTPS requests
- The max-age value must be minimally 18 weeks
- includeSubDomains directive is mandatory when subdomains are used
- It should specify preload directive

Parameters:

- max-age – time (in seconds) for which the client should remember
- includeSubDomains – specifies that the rule applies to all subdomains
- preload – request for inclusion into the preload list of HTTPS only sites[74]

4.2.2 Content-Security-Policy

Content Security Policy prevents cross-site-scripting (XSS), Clickjacking, code injection attacks by whitelisting allowed content on a website. When content is not whitelisted, a browser will block it. It should be understood as a last line of defense as it depends on browser support. This security header has an option to report a violation to preset URI.

Content-Security-Policy set to "object-src 'self'" was tested with the Content Server as part of this work. This header allows only source for the <object>, <embed>, and <applet> elements from the same origin.[75]

4.2.3 Certificate Transparency (Expect-CT)

The Certificate Transparency (CT) monitoring allows the website owner to quickly and easily become aware of new certificates issued and make sure they are genuine. It is a way to mitigate the possibility of a "rough" Certificate Authority issuing a certificate for an attacker to a legitimate website. A new certificate is submitted to public Certificate Transparency logs. Additionally, this certificate is extended with Signed Certificate Timestamps (SCT) from Certificate Transparency logs. A certificate without SCT is not considered valid for a browser if sending host is on the browser's Known CT servers list.

Certificate transparency is defined in RFC6962.[76]

Parameters:

- report-only (optional) – reports missing valid Signed Certificate Timestamps to a URI specified by report-uri parameter and instruct a browser to continue with the connection.
- enforce (optional) – instructs a browser to enforce compliance with Certificate Transparency and refuse future connections violating Certificate Transparency policy until the time set by max-age parameter is reached.
- report-uri (optional) – URI where a browser reports a violation of the policy, the only way to test the functionality of this policy would be to purchase a certificate from a Certification Authority without embedded Signed Certificate Timestamps and then serve it to a browser.
- max-age (required) – the number of seconds after a browser receives the Expect-CT header; until this time passes, the browser should recognize the host from whom the message was received as a Known Expect-CT host.[77]

4.2.4 Referrer-Policy

The Referrer-policy header controls how much referrer information may be included with requests.[78]

Possible policy values:

- no-referrer – sends no referrer information
- no-referrer-when-downgrade – does not send referrer information if the protocol is downgraded as HTTPS to HTTP
- unsafe-url – full URL will be sent regardless of security

- same-origin – referrer will be sent only for the same-origin site, not for cross-origin
- strict-origin – sends referrer only for HTTPS
- strict-origin-when-cross-origin – the full URL will be sent only when strict protocol (e.g., HTTPS) is used
- origin – send only site info, not a specific URL in all the requests
- origin-when-cross-origin – send a specific URL on the same-origin, but only site info for cross-origin and when the protocol is downgraded [79]

Content Server and OTDS use referrer info heavily. The Content Server platform was successfully tested with Referrer-policy set to same-origin. However, limiting referrer info may disrupt integrations using frames and iFrames.

4.2.5 X-Content-Type-Options

Instructs a browser to consider file types as defined and disallows content sniffing because some MIME types can be executable.

The only parameter of this header is "nosniff".[80]

NOTE: There was an issue with Blazon Publish step with this security header during this work. It is an issue only in SmartUI. Classic UI works without any issue. This issue was resolved by patch pat162003342.

4.2.6 X-Permitted-Cross-Domain-Policies

This policy is used to allow cross-domain requests from PDF and Adobe Flash Player files. It is safer to prohibit this functionality unless it is needed. Whitelisting allowed cross-domains is done in an XML file "crossdomain.xml" located in the website's root directory.

Possible policy values:

- none – no policy is allowed
- master-only – allow only the master policy

- all – everything is allowed
- by-content-only – allow only a particular type of content. Example – XML
- by-ftp-only – applicable only for an FTP server[81]

The Content Server platform was successfully tested with this policy set to "none".

4.2.7 X-XSS-Protection

This policy prevents some level of Cross-Site-Scripting attacks by enabling an XSS filter in a browser. This filter is based on blacklisting characters and tags in request parameters.

Possible parameter values:

- 0 – the filter is disabled
- 1 – the filter is enabled and sanitize the page if an attack is detected
- `1;mode=block` – the filter is enabled, and rendering of the page is blocked
- `1;report=http://example.com/report_URI` – the filter is enabled, and an attack is reported[82]

The Content Server platform was successfully tested with this policy set to "1;mode=block".

4.2.8 X-Frame-Option

The security header X-Frame-Options helps to prevent Clickjacking vulnerability. The browser will block embedding the site in frame or iframe.

Possible parameter values:

- SAMEORIGIN – Frame or iframe is only allowed from the same site.
- DENY – Prevents any embedding using frame or iframe.
- ALLOW-FROM – Allows framing only on a particular URL.[83]

The Content Server has its security parameter limiting embedding. It is located in the Admin pages (`?func=admin.securityvars`). The checkbox *Prevent request handlers from being embedded in external frames* must be selected to enable it.

OTDS has a hidden system attribute called X-Frame-Options Header. OTDS sends X-Frame-Options header with the value "SAMEORIGIN" for the login page only. The header is not sent with the web administration pages.

The Archive Server can be used with this policy set to "DENY".

4.3 Complicating Footprinting

During the *Footprinting* (CAPEC-169)[84] stage, an attacker tries to find information about running services. Knowing which software runs on an opened port helps find out a proper exploit. Furthermore, knowing the exact version of the service can speed up this process significantly.

Apache Tomcat in the default configuration clearly shows version info on the landing page. Additionally, error messages and default web applications installed with Apache Tomcat inform about the version. However, the default applications are not necessary for a production run of Apache Tomcat, and they can be removed. On the other hand, standard HTTP error codes such as 404 or 500 should not be removed. Instead, they may be replaced by sanitized version. Alternatively, a configuration of RemoteIPValve may limit the IP addresses to which they are sent.[85]

5 ELIMINATION OF THREATS

The following chapters contain exact steps to lower the attack vector of the Content Server platform. They use these conventions:

- `%SystemRoot%` – the percentage characters `%` are used to denote system variable.
- `<site>` – the brackets `<>` are used to mark a variable or placeholder. It must be replaced by the value correct in a target environment.

For better readability, the following variables are used to define the root installation folder for respective components:

- `%OTHOME%` – represents the root folder of the Content Server installation
- `%CATALINA_HOME%` – the Apache Tomcat installation root folder
- `%OTDS_HOME%` – the root folder of OTDS installation

5.1 Preventing Apache Tomcat Information Disclosure

5.1.1 Changing the Default Landing Page

1. Remove the content of `%CATALINA_HOME%\webapps\ROOT`
2. Add a simple index.jsp (the example of its content can be found in Appendix A II.) and favicon files.

5.1.2 Removing Server Version Information

1. Create following the folder structure: `%CATALINA_HOME%\lib\org\apache\catalina\util`
2. Create ServerInfo.properties file with this content: server.info=

NOTE: Instead of removing version info for all requests, it is possible to limit this info to specified external IP addresses using Remote IP Valve in Apache Tomcat.[85]

5.1.3 Removing Default Web Applications

In the `%CATALINA_HOME%\webapps` folder, delete the following subfolders:

- docs
- manager
- examples (if installed)
- host-manager (if installed)

5.2 Apache Tomcat – Enable HSTS and Other Security Headers

Uncomment or add the following lines into the web-app node of `%CATALINA_HOME%\conf\web.xml` file:

```
<filter>
<filter-name>httpHeaderSecurity</filter-name>
<filter-class>org.apache.catalina.filters.HttpHeaderSecurityFilter
</filter-class>
<async-supported>true</async-supported>
<init-param>
<param-name>hstsEnabled</param-name>
<param-value>true</param-value>
</init-param>
<init-param>
<param-name>hstsMaxAgeSeconds</param-name>
<param-value>31536000</param-value>
</init-param>
<init-param>
<param-name>hstsIncludeSubDomains</param-name>
<param-value>true</param-value>
</init-param>
<init-param>
<param-name>antiClickJackingEnabled</param-name>
<param-value>false</param-value>
</init-param>
<init-param>
<param-name>xssProtectionEnabled</param-name>
<param-value>true</param-value>
</init-param>
<init-param>
<param-name>blockContentTypeSniffingEnabled</param-name>
<param-value>true</param-value>
</init-param>
</filter>
<filter-mapping>
<filter-name>httpHeaderSecurity</filter-name>
<url-pattern>/*</url-pattern>
<dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

Possible parameter values:

- **hstsEnabled** – must be set to true to enable HSTS
- **hstsMaxAgeSeconds** – time in seconds

  NOTE: It is better to start with a low number (e.g., 300) until a whole platform is tested and works properly. When tested, it is recommended to set it to 31536000. However, there was no occurred during testing of the Content Server platform.
- **antiClickJackingEnabled** – enables security header X-Frame-Options

NOTE: This security header affects the possibility to load Content Server or OTDS as frame or iFrame from other systems.

- **antiClickJackingOption** – defines which option is sent with X-Frame-Options header. They may be:
  - **DENY** – default value with antiClickJackingEnabled, no need to be specified. It prevents any domain from embedding content using frame or iframe.
  - **SAMEORIGIN** – allows frame or iframe of content only from the same site origin
  - **ALLOW-FROM** – allows frame or iframe a content only on a particular URI
- **antiClickJackingUri** – if ALLOW-FROM value is used with parameter antiClickJackingOption, this parameter specifies URI(s), which can embed a web page in frame or iframe. This option has limitations in browser support.
- **xssProtectionEnabled** – add security header X-XSS-Protection with value "1;mode=block"
- **blockContentTypeSniffingEnabled** – adds security header X-Content-Type-Options with parameter "nosniff"

As mentioned in chapter 4.2.8 Content Server and OTDS offer their functionality to limit the embedding of Content Server or OTDS in other websites. For this reason, this security header X-Frame-Option was not configured and tested for Apache Tomcat running OTDS. However, it is a valid option for Apache Tomcat running Archive Server.

5.3 Apache Tomcat – Disabling AJP Connector

Default Apache Tomcat configuration has enabled AJP Connector. However, AJP Connector is not needed for OTDS and Archive Server. It may be used only in the Active/Active Archive Server cluster for the load balancer when Apache mod_jk is used. Access to the AJP Connector port must be limited to only other machines in the cluster in such a case. An upgrade of Apache Tomcat to a version higher than 8.5.51 or 9.0.31 is recommended.

To disable AJP Connector the following node must be deleted or commented out in `%CATALINA_HOME%\conf\server.xml`:

```
<Connector protocol="AJP/1.3" address="::1" port="8009"
redirectPort="8443"
/>
```

5.4 Apache Tomcat – Disabling Shutdown Port

Edit `%CATALINA_HOME%\conf\server.xml` and in following node replace port number with -1: `<Server port="8005" shutdown="SHUTDOWN">`

5.5 Apache Tomcat – Enabling Secure and HTTP Only Cookies

Secure cookies can be enabled directly in OTDS and the Content Server. However, it may be safer to enable HTTP Only cookies in Apache Tomcat too:

1. Edit the `%OTHOME%\config\opentext.ini` and, in the section [options], add the parameter: wantSecureCookies=TRUE
2. For systems using OTDS, these settings must also be set in OTDS. To do this, set otds.as.wantSecureCookies parameter from the OTDS admin page.
3. Add to `%CATALINA_HOME%\conf\web.xml` in session-config section:

   ```
   <cookie-config>
   <http-only>true</http-only>
   </cookie-config>
   ```

Note: When a proxy server or load balancer is used, HTTP-Only cookies have to be tested. A load balance or proxy server may terminate TLS, and OTDS will be accessed over HTTP. In such a case, the proxy or load balancer has to set the flag itself or send X-Forwarded-Proto header to OTDS. Furthermore, Apache Tomcat's RemoteIpValve functionality must be set by adding the following line into server.xml:

```
<Valve className='org.apache.catalina.valves.RemoteIpValve'
protocolHeader='x-forwarded-proto'/>
```

5.6 Apache Tomcat – Disabling Weak Transport Protocols and Ciphers

To limit used transport protocols, the HTTPS Connector node of `%CATALINA_HOME%\conf\server.xml` must be modified.

Transport protocols TLSv1.2 and TLSv1.3 are set using:

sslProtocol="TLSv1.2" sslEnabledProtocols="TLSv1.2+TLSv1.3"

Additionally, limit used cipher suits and their order by adding:

```
ciphers="TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256"
SSLHonorCipherOrder="true"
```

Complete connector node should look like this example:

```
<Connector port="8090" maxHttpHeaderSize="8192"
maxThreads="250" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true" acceptCount="100"
scheme="https" secure="true" SSLEnabled="true" clientAuth="false"
sslProtocol="TLSv1.2" sslEnabledProtocols="TLSv1.2+TLSv1.3"
keyAlias="<server>" keystoreFile="<keystore file path>"
keystorePass="<password>"
ciphers="TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256"
SSLHonorCipherOrder="true"
 />
```

NOTE: TLSv1.3 would not work without Java upgraded to OpenJDK11.

5.7 OpenDJ – Preventing Information Disclosure

Sending the notice of disconnection needs to be disabled in LDAP and LDAPS connection handlers. It is done in the command-line interface using the following commands:

`%OTDS_HOME%\opendj\bat\dsconfig` set-connection-handler-prop
--handler-name "LDAP Connection Handler" --set send-rejection-notice:false

`%OTDS_HOME%\opendj\bat\dsconfig` set-connection-handler-prop
--handler-name "LDAPS Connection Handler" --set send-rejection-notice:false

NOTE: The administrator will be asked to provide additional info: server name, administration port, administrator username, and password.

NOTE: The send-rejection-notice property is an advanced property, and it is not shown in the interactive mode of the dsconfig command. However, running dsconfig with –advanced option will show it.

5.8 OpenDJ – Disabling Weak Transport Protocols and Ciphers

Disabling weak transport protocols and ciphers must be done for Admin, Replication ports, and connection handlers: LDAP and LDAPS.

Available protocols and ciphers can be listed using following command for Admin port:

```
%OTDS_HOME%\opendj\bat\ldapsearch --port 4440 --bindDN "cn=Directory Manager"
--baseDN "" --trustAll --useSSL
--searchScope base "(&)" supportedTLSCiphers supportedTLSProtocols
```

A similar way information about protocols and ciphers can be acquired for:

- Replication port: 8989 (default value)
- LDAP: 389 (default value)
- LDAPS: 636 (default value)

For example, to enable specific protocol and cipher suites following command may be used:

```
%OTDS_HOME%\opendj\bat\dsconfig set-crypto-manager-prop --port 4440
--bindDN "cn=Directory Manager"
--add ssl-cipher-suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
--add ssl-cipher-suite:TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 --add ssl-protocol:TLSv1.2
--trustAll --no-prompt
```

A list of recommended strong cipher suites for OpenDJ was not created as part of this work as it depends on used directory server.

NOTE: LDAPS Connection Handler is disabled by default after installation and must be enabled.

NOTE: Default port numbers are used in the previous commands. It must be modified if other port numbers were used.

5.9 OpenDJ – SASL Security Layer Excessive Memory Use

If possible, TLS protocol should be used instead of SASL security layers. DIGEST-MD5 mechanism is already obsolete.[86]

Set DIGEST-MD5 and GSS-API mechanisms quality-of-protection properties to "none" to prevent security layer negotiation. The following commands will do it:

`%OTDS_HOME%\opendj\bat\dsconfig` set-sasl-mechanism-handler-prop --handler-name DIGEST-MD5 --set quality-of-protection:none

`%OTDS_HOME%\opendj\bat\dsconfig` set-sasl-mechanism-handler-prop --handler-name GSS-API --set quality-of-protection:none

NOTE: GSS-API is not enabled by default in OpenDJ.

5.10 IIS – Security Headers

IIS allows setting more types of security headers than Apache Tomcat due to its more flexible configuration.

The easiest way to set Security headers in IIS is by using the appcmd command.

Some of the security headers design supports reporting of problematic requests or certificates for logging. There is a possibility to set attribute report-uri. A portal runing on URL `https://report-uri.com` allows collecting those reports up to 10 thousand per month for free. The report-uri parameters are pointing to this service in the following commands. However, they may be omitted if not required.

Content-Security-Policy

`%SystemRoot%\System32\inetsrv\appcmd.exe` set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='Content-Security-Policy',value='object-src self']"

NOTE: Unfortunately, during this work a viable solution how to pass apostrophes to appcmd command was not discovered. As workaround apostrophes were added around keyword 'self' using IIS Manager.

Test settings for Certificate Transparency

```
%SystemRoot%\System32\inetsrv\appcmd.exe set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='Expect-CT',value='report-only, max-age=300',
report-uri="https://<subdomain>.report-uri.com/r/d/ct/reportOnly"]"
```

Production settings for Certificate Transparency

```
%SystemRoot%\System32\inetsrv\appcmd.exe set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='Expect-CT',value='enforce, max-age=604800',
report-uri="https://<subdomain>.report-uri.com/r/d/ct/enforce"]"
```

Referrer-Policy

```
%SystemRoot%\System32\inetsrv\appcmd.exe set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='Referrer-Policy',value='same-origin']"
```

Strict-Transport-Security (HSTS)

```
%SystemRoot%\System32\inetsrv\appcmd.exe set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='Strict-Transport-Security',value='max-age=31536000;
includeSubDomains']"
```

X-Content-Type-Options

```
%SystemRoot%\System32\inetsrv\appcmd.exe set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='X-Content-Type-Options',value='nosniff']"
```

X-Permitted-Cross-Domain-Policies

```
%SystemRoot%\System32\inetsrv\appcmd.exe set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='X-Permitted-Cross-Domain-Policies',value='none']"
```

X-XSS-Protection

`%SystemRoot%\System32\inetsrv\appcmd.exe` set config "<site>"
-section:system.webServer/httpProtocol
/+"customHeaders.[name='X-XSS-Protection',value='1;mode=block']"

5.11 IIS – Disabling Weak Transport Protocols and Ciphers

Disabling or enabling of the transport protocols or ciphers for IIS is done in the Registry. After modifying registry values whole Operating System must be rebooted.

The protocols are enabled or disabled by creating appropriate subkeys in the key:
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\`
`Protocols`

Each protocol has a subkey with Server and Client settings. They have an appropriate DWORD value of Enabled or DisabledbyDefault. Their values are set to 0x0 for disabling. Alternatively, 0xffffffff for enabling. Be aware that there may be exceptions to this rule instead of 0xffffffff, must be used 0x00000001. If a key or a value is missing default value is used.

The ciphers are enabled or disabled in the key:
`HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\SCHANNEL\`
`Ciphers`

The same system as for protocols is used to disable or enable cipher.

A complete list of tested registry keys is can be found in Appendix A III.[87]

Removing Client TLSv1.0 protocol caused an issue during redirection from the OTDS server after authentication back to the Content Server. Also, the removal following cipher suites generated the same issue:

- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA

It was unclear if the configuration of the test platform caused this issue. Further investigation into this issue will be beneficial.

CONCLUSION

In a hypothetical scenario, where an attacker tasks an OpenText expert to extract valuable data from the Content Server platform, the expert will focus on OTDS. It is often located in DMZ due to authentication redirections between the Content Server and OTDS if the attacker provides him remote shell on the OTDS machine. First, he will reset the *Directory Manager* password. The next step will be to create an user with a name that looks like it belongs to a technical administration group. Add this user to the *otds.admin* group. After that, the extraction of information may start. To improve this procedure, he may modify interests for the *Audit log* by excluding this user before the extraction or clean up all records for this user afterward.

If the task is laterally moving within the platform and acquire persistence, the focus will be on the System Center. It is usually located with backend servers. So, it may be harder for our attacker to get a remote shell. Now, he can start listening for incoming communication on port 8888. Until a legitimate administrator logs in over HTTP and the password is captured. The default HTTP protocol for the System Center is rarely reconfigured to HTTPS. Probably it is a backend server, and so it is "safe." With this password, it is only a matter of crafting obfuscated PowerShell script to avoid detecting anti-malware technology used on target machines. The next step is to create and run the *Execution plan* with this script on all machines with the System Center Agent. The last step will be creating a *servtab* file for the Archive Server or an OScript patch file for the Content Server, starting malicious code to gain persistence.

What weaknesses of the OpenText Content Server platform were identified?

- The Archive Server stores its database credentials in encrypted form within a plaintext file. However, the encryption algorithm does not use salt. Therefore, for the same input, the same encrypted output can be created on any other Archive Server machine. So, a rainbow table can be precomputed to crack the credentials.
- The Archive Server, when used with built-in user management, has default empty password administrator user *dsadmin*.
- The Content Server does not implement a robust mechanism for preventing manipulation with the Audit log directly in a database.
- The Content Server platform may be used to disseminate infected MS Office files

or malicious links. There is no standard integration for scanning uploaded files with an anti-malware solution.

- At the start of the Content Server, OScript patch files are executed without any mechanisms to prevent the execution of malicious code.
- The XSS exploit is publicly available for the Content Server version 20.3.
- The default configuration of OTDS and Apache Tomcat does not restrict manipulation with cookies using JavaScript.
- The default configuration of Apache Tomcat has enabled the AJP Connector, which contains the exploitable vulnerability.
- The OpenDJ Directory Manager password may be reset without losing any data with local access to the OTDS machine.
- The default OpenDJ configuration uses weak transport protocols and cipher suites.
- The OpenDJ version contains a vulnerability, which may be used to deplete memory.
- The default OpenDJ configuration allows an attacker to get information about the expected protocol and the qualified Java class name of the application.
- An attacker may create a *servtab* file starting malicious code. Copying this file into the proper folder of the Archive Server will start it under the service user used to run the Spawner service.

A summary of recommendations for improving the security of the Content Server platform:

- Configure HTTPS where possible (IIS, Tomcat, OpenDJ, and Eclipse Jetty) and disable weak transport protocols and cipher suits as described in this thesis. It does not mean that all communication between components in the backend must be done over HTTPS. However, all administrations must be done over HTTPS without exceptions.
- Configure HTTP Security Headers, where possible. It may serve as the last barrier to prevent an attack when an attacker discovers unknown XSS or gets into an end-user machine and sets the attacker's certificate as trusted for MITM.

Information in this work will help with initial settings. However, due to different integrations in the target system, proper regression testing must be done.

- Tomcat hardening: prevent information disclosure, disable the shutdown port, disable the AJP Connector
- Enable Secure and 'HttpOnly' Cookies in OTDS and Apache Tomcat
- Change the OpenDJ Directory Manager password immediately after OTDS installation
- Install and configure PVA Virus Scan Integration Module for AV checking of uploaded documents
- If possible, deploy additional protection for OTDS and System Center machines. For example, by implementing *Application Whitelisting.*
- Uninstall or disable any module or component, which is not going to be used soon.

All these recommendations are easily implemented. In addition, most of them do not bring additional license costs. Additionally, chapter 5 includes detailed and tested implementation steps, which will save some implementation time.

A significant number of organizations allow access to their Content Server via the Internet. Mainly for sharing documents with their external partners. Enabling *Secure and HTTP Only Cookies* is a crucial step. Using the Content Server platform via the Internet without adequately secured cookies is a high risk. An attacker can use a wide range of techniques to steal end-user cookies and use his identity to access the system. Furthermore, the addition of HSTS to the scenario mentioned above prevents some types of MITM attacks. Implementing these two recommendations is a bare minimum.

Other recommended HTTP Security Headers, disabling weak transport protocols and ciphers, limiting all administration connection to HTTPS-only should follow as they prevent a wide variety of common attacks. They should be understood as part of the best practice during the setup of the platform.

Finally, hardening Apache Tomcat and OpenDJ may have the lowest impact from the recommendations. However, we may be surprised in the future. We learn about attacks on well-established companies using quite innovative techniques abusing forgotten or underestimated weaknesses from time to time.

There are undoubtedly blind spots in the results of this work. However, the implementation of all proposed changes will reduce the platform attack vector significantly. Moreover, indeed, it will lead to more admirable PEN testing results. Furthermore, the "set and forget" approach will not work if the goal is increased security. Therefore, a process of continual improvements and testing needs to be set up.

This analysis opened many questions and ideas for future research:

- Create a more formal methodology for security analysis of enterprise systems than brainstorming or general checklists. Second, incorporate a more robust solution for measuring attack surface and weaknesses. Third, include the use of Agile Threat Modeling Toolkit or similar solution for data flow analysis from a security perspective.
- Create a comprehensive hardening guide for the Content Server platform with tested combinations of different components, build-in security settings of OpenText components, and third-party components included in OpenText products.
- Create a guide on setting up Apache Tomcat Security Manager to run OTDS and the Archive Server?
- Detailed analysis of known third-party components known vulnerabilities and weaknesses. May they be ignored due to the way how they are integrated into the platform?
- Is it possible to abuse known vulnerabilities of Eclipse Jetty to bypass authentication?
- Research more OpenDJ and OTDS integration. For example, is there a possibility to get user passwords from OpenDJ? Or is it possible to create in OpenDJ a user who is not visible in OTDS Administration?
- What exactly happens when a browser gets the same security header twice? Each instance with a different protection level. Does it lower protection?
- Investigate if it is possible to directly connect to the H2 database of the System Center with local access to the machine with the System Center Manager.
- Is there a way to circumvent built-in user management of the System Center?

The discovered weaknesses of the OpenText Content Server platform and their solutions will improve the platform's security as this knowledge will be shared among Content

Server professionals. However, this research also opens the question of how to assess the attack surface of complex systems. Especially point out that focusing only on attack surface in the context of *barriers, methods, flows*, or *reachable vulnerabilities* is not sufficient nowadays. Cybersecurity of enterprise information systems needs to focus on *feature* and *adversaries* contexts of attack surface as well.

REFERENCES

[1] Michael Woodbridge; Marko Sillanpaa; Lane Severson: Magic Quadrant for Content Services Platforms. November 2020, iD G00464767.
URL `https://www.gartner.com/en/documents/3993178/magic-quadrant-for-content-services-platforms`

[2] Theisen, C.; Munaiah, N.; Al-Zyoud, M.; et al.: Attack surface definitions: A systematic literature review. *Information and Software Technology*, volume 104, December 2018: pp. 94–103, ISSN 09505849, doi:10.1016/j.infsof.2018.07.008.
URL `https://linkinghub.elsevier.com/retrieve/pii/S0950584918301514`

[3] Application Threat Modeling.
URL `https://owasp.org/www-community/Application_Threat_Modeling`

[4] Manadhata, P. K.; Karabulut, Y.; Wing, J. M.: Measuring the attack surfaces of sap business applications. In *IEEE International Symposium on Software Reliability Engineering*, 2008.

[5] Support Lifecycle. Publication Title: OpenText My Support.
URL `https://knowledge.opentext.com/knowledge/cs.dll/open/lifecycle`

[6] Content Server Application Security Hardening Guide. December 2016, publication Title: OpenText My Support.
URL `https://knowledge.opentext.com/go/65175979`

[7] Security Capabilities Overview for OpenText™ Content Suite Platform. September 2016, publication Title: OpenText My Support.
URL `https://knowledge.opentext.com/knowledge/llisapi.dll/Open/63972286`

[8] OTDS - OpenDJ Backend Security Hardening of Ports and Certificates.
URL `https://knowledge.opentext.com/knowledge/cs.dll/kcs/kbarticle/view/KB13827329`

[9] Content Server - OTDS - System Center - Tempo Box - How to implement custom generated Certificates for OpenText products.
URL `https://knowledge.opentext.com/knowledge/cs.dll/kcs/kbarticle/view/KB4957564`

[10] Document Encryption Open Text Archive Server. April 2009, publication Title: OpenText My Support.

URL `https://knowledge.opentext.com/knowledge/cs.dll/Open/1142249703_817`

[11] Secure Archiving. March 2010, publication Title: OpenText My Support.
URL `https://knowledge.opentext.com/knowledge/cs.dll/Open/0976813282_463_pdf`

[12] CVSS v3.1 User Guide. Publication Title: FIRST — Forum of Incident Response and Security Teams.
URL `https://web.archive.org/web/20210223173640/https://www.first.org/cvss/v3.1/user-guide`

[13] Software vulnerabilities.
URL `https://encyclopedia.kaspersky.com/knowledge/software-vulnerabilities/`

[14] CWE - Common Weakness Scoring System (CWSS).
URL `https://cwe.mitre.org/cwss/cwss_v1.0.1.html`

[15] CWSS calculator.
URL `https://cwss-score.info/`

[16] Hansman, S.; Hunt, R.: A taxonomy of network and computer attacks. *Computers & Security*, volume 24, no. 1, February 2005: pp. 31–43, ISSN 01674048, doi: 10.1016/j.cose.2004.06.011.
URL `https://linkinghub.elsevier.com/retrieve/pii/S0167404804001804`

[17] CAPEC - CAPEC List Version 3.4.
URL `https://capec.mitre.org/data/index.html`

[18] Steal Web Session Cookie, Technique T1539 - Enterprise \textbar MITRE ATT&CK®.
URL `https://attack.mitre.org/techniques/T1539/`

[19] Mohd. Ehmer Khan; Khan, F.: Comparative Study of White Box, Black Box and Grey Box Testing Techniques. *International Journal of Advanced Computer Science and Applications*, volume 3, no. 6, 2012: pp. 12–15, ISSN 2156-5570, doi:10.14569/issn.2156-5570.
URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.685.1887&rep=rep1&type=pdf#page=22`

[20] AIIM: What is Document Management (DMS)?
URL `https://www.aiim.org/what-is-document-imaging`

[21] AIIM: What is ECM?
URL https://info.aiim.org/what-is-ecm

[22] OpenText™ Content Server Product Overview. April 2016.
URL https://www.opentext.com/file_source/OpenText/en_US/PDF/opentext-product-overview-co-opentext-content-server-en.pdf

[23] CAPEC - CAPEC-70: Try Common or Default Usernames and Passwords (Version 3.4).
URL https://capec.mitre.org/data/definitions/70.html

[24] CAPEC - CAPEC-565: Password Spraying (Version 3.4).
URL https://capec.mitre.org/data/definitions/565.html

[25] CAPEC - CAPEC-94: Man in the Middle Attack (Version 3.4).
URL https://capec.mitre.org/data/definitions/94.html

[26] CAPEC - CAPEC-227: Sustained Client Engagement (Version 3.4).
URL https://capec.mitre.org/data/definitions/227.html

[27] CAPEC - CAPEC-55: Rainbow Table Password Cracking (Version 3.4).
URL https://capec.mitre.org/data/definitions/55.html

[28] CAPEC - CAPEC-268: Audit Log Manipulation (Version 3.4).
URL https://capec.mitre.org/data/definitions/268.html

[29] User Execution, Technique T1204 - Enterprise - MITRE ATT&CK®.
URL https://attack.mitre.org/techniques/T1204/

[30] Internal Spearphishing, Technique T1534 - Enterprise - MITRE ATT&CK®.
URL https://attack.mitre.org/techniques/T1534/

[31] PVA Virus Scan Integration Module. Publication Title:.
URL https://www.opentext.com/products-and-solutions/partners-and-alliances/partner-solutions-catalog/partner-solutions-catalog-detail?id=a10D0000000kyQPIAY

[32] CVE - Search Results.
URL https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=OpenText

[33] CVE - CVE-2021-3010.
URL https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-3010

[34] Breński, K.: OpenText Content Server 20.3 - 'multiple' Stored Cross-Site Scripting. February 2021, publication Title: Exploit Database.
URL `https://www.exploit-db.com/exploits/49578`

[35] OWASP AntiSamy.
URL `https://owasp.org/www-project-antisamy/`

[36] OpenDJ - Open Directory Server · Open Identity Platform. Publication Title: Open Identity Platform - Open Source Solutions for Access Management, Identity Management, Directory Services.
URL `https://www.openidentityplatform.org/opendj`

[37] CWE - CWE-497: Exposure of Sensitive System Information to an Unauthorized Control Sphere (4.4).
URL `https://cwe.mitre.org/data/definitions/497.html`

[38] CAPEC - CAPEC-170: Web Application Fingerprinting (Version 3.4).
URL `https://capec.mitre.org/data/definitions/170.html`

[39] CWE - CWE-326: Inadequate Encryption Strength (4.4).
URL `https://cwe.mitre.org/data/definitions/326.html`

[40] CWE - CWE-1004: Sensitive Cookie Without 'HttpOnly' Flag (4.4).
URL `https://cwe.mitre.org/data/definitions/1004.html`

[41] CWE - CWE-640: Weak Password Recovery Mechanism for Forgotten Password (4.4).
URL `https://cwe.mitre.org/data/definitions/640.html`

[42] How does DS (All versions) store password values? - Knowledge - BackStage.
URL `https://backstage.forgerock.com/knowledge/kb/article/a44757687`

[43] Security Advisories - Knowledge - BackStage.
URL `https://backstage.forgerock.com/knowledge/kb/book/b21824339#a47486518`

[44] OpenDJ Security Advisory #201703 - Knowledge - BackStage.
URL `https://backstage.forgerock.com/knowledge/kb/article/a38293619`

[45] CAPEC - CAPEC-541: Application Fingerprinting (Version 3.4).
URL `https://capec.mitre.org/data/definitions/541.html`

[46] DS/OpenDJ Security Advisory #201706 - Knowledge - BackStage.
URL `https://backstage.forgerock.com/knowledge/kb/article/a27262510`

[47] CAPEC - CAPEC-603: Blockage (Version 3.4).
URL `https://capec.mitre.org/data/definitions/603.html`

[48] POODLE SSL Vulnerability and OpenDJ - Knowledge - BackStage.
URL `https://backstage.forgerock.com/knowledge/kb/article/a29539600`

[49] CVE - CVE-2014-3566.
URL `https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3566`

[50] Third party software licenses. December 2020, publication Title: OpenText My Support.
URL `https://knowledge.opentext.com/knowledge/cs.dll/Open/68246877`

[51] OpenText™ Archive Center Administration Guide. May 2017, publication Title: OpenText My Support.
URL `https://knowledge.opentext.com/knowledge/llisapi.dll/Open/67813205`

[52] CWE - CWE-261: Weak Encoding for Password (4.4).
URL `https://cwe.mitre.org/data/definitions/261.html`

[53] CAPEC - CAPEC-478: Modification of Windows Service Configuration (Version 3.4).
URL `https://capec.mitre.org/data/definitions/478.html`

[54] CWE - CWE-521: Weak Password Requirements (4.4).
URL `https://cwe.mitre.org/data/definitions/521.html`

[55] Indirect Command Execution, Technique T1202 - Enterprise - MITRE ATT&CK®.
URL `https://attack.mitre.org/techniques/T1202/`

[56] Lateral Tool Transfer, Technique T1570 - Enterprise - MITRE ATT&CK®.
URL `https://attack.mitre.org/techniques/T1570/`

[57] Command and Scripting Interpreter: PowerShell, Sub-technique T1059.001 - Enterprise - MITRE ATT&CK®.
URL `https://attack.mitre.org/techniques/T1059/001/`

[58] CAPEC - CAPEC-524: Rogue Integration Procedures (Version 3.4).
URL `https://capec.mitre.org/data/definitions/524.html`

[59] CAPEC - CAPEC-523: Malicious Software Implanted (Version 3.4).
URL `https://capec.mitre.org/data/definitions/523.html`

[60] Eclipse Jetty : List of security vulnerabilities.
URL `https://www.cvedetails.com/vulnerability-list/vendor_id-10410/product_id-34824/Eclipse-Jetty.html`

[61] CWE - CWE-200: Exposure of Sensitive Information to an Unauthorized Actor (4.4).
URL `https://cwe.mitre.org/data/definitions/200.html`

[62] CWE - CWE-444: Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling') (4.4).
URL `https://cwe.mitre.org/data/definitions/444.html`

[63] CWE - CWE-59: Improper Link Resolution Before File Access ('Link Following') (4.4).
URL `https://cwe.mitre.org/data/definitions/59.html`

[64] Manadhata, P. K.; Wing, J. M.: An Attack Surface Metric. *IEEE Transactions on Software Engineering*, volume 37, no. 3, May 2011: pp. 371–386, ISSN 0098-5589, doi:10.1109/TSE.2010.60.
URL `http://ieeexplore.ieee.org/document/5482589/`

[65] CWE - CWE-757: Selection of Less-Secure Algorithm During Negotiation ('Algorithm Downgrade') (4.4).
URL `https://cwe.mitre.org/data/definitions/757.html`

[66] CAPEC - CAPEC-158: Sniffing Network Traffic (Version 3.4).
URL `https://capec.mitre.org/data/definitions/158.html`

[67] Moriarty, K.; Farrell, S.: Deprecating TLS 1.0 and TLS 1.1. 2021, ISSN 2070-1721.
URL `https://www.rfc-editor.org/info/rfc8996`

[68] Barnes, R.; Thomson, M.; Pironti, A.; et al.: Deprecating Secure Sockets Layer Version 3.0. Technical report RFC7568, RFC Editor, june 2015, doi:10.17487/RFC7568.
URL `https://www.rfc-editor.org/info/rfc7568`

[69] Popov, A.: Prohibiting RC4 Cipher Suites. 2015, ISSN 2070-1721.
URL `https://www.rfc-editor.org/info/rfc7465`

[70] MD5 collisions and the impact on computer forensics. *Digital Investigation*, volume 2, no. 1, February 2005: pp. 36–40, ISSN 1742-2876, doi:10.1016/j.diin.2005.01.004.

URL https://www.sciencedirect.com/science/article/abs/pii/S1742287605000058

[71] Sweet32: Birthday attacks on 64-bit block ciphers in TLS and OpenVPN.
URL https://sweet32.info/

[72] lastnameholiu; Kent Sharkey; David Coulter; et al.: TLS Cipher Suites in Windows 10 v1607 - Win32 apps.
URL https://docs.microsoft.com/en-us/windows/win32/secauthn/tls-cipher-suites-in-windows-10-v1607

[73] Hodges, J.; Jackson, C.; Barth, A.: HTTP Strict Transport Security (HSTS). 2012, ISSN 2070-1721.
URL https://www.rfc-editor.org/info/rfc6797

[74] HSTS Preload List Submission.
URL https://hstspreload.org/

[75] Content-Security-Policy - HTTP.
URL https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy

[76] Laurie, B.; Langley, A.; Kasper, E.: Certificate Transparency. 2013, ISSN 2070-1721.
URL https://www.rfc-editor.org/info/rfc6962

[77] How CT Works : Certificate Transparency.
URL https://certificate.transparency.dev/howctworks/

[78] Referer and Referrer-Policy best practices. Publication Title: web.dev.
URL https://web.dev/referrer-best-practices/

[79] Referrer-Policy - HTTP.
URL https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy

[80] X-Content-Type-Options - HTTP.
URL https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options

[81] OWASP Secure Headers Project - OWASP.
URL https://wiki.owasp.org/index.php/OWASP_Secure_Headers_Project#xpcdp

[82] Understanding XSS Auditor. October 2017, publication Title: Virtue Security.
URL `https://www.virtuesecurity.com/understanding-xss-auditor/`

[83] X-Frame-Options - HTTP.
URL `https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options`

[84] CAPEC - CAPEC-169: Footprinting (Version 3.4).
URL `https://capec.mitre.org/data/definitions/169.html`

[85] Apache Tomcat 8 Configuration Reference (8.5.66) - The Valve Component.
URL `https://tomcat.apache.org/tomcat-8.5-doc/config/valve.html#Remote_IP_Valve`

[86] Simple Authentication and Security Layer (SASL) Mechanisms.
URL `https://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xhtml`

[87] Deland-Han: Restrict cryptographic algorithms and protocols - Windows Server.
URL `https://docs.microsoft.com/en-us/troubleshoot/windows-server/windows-security/restrict-cryptographic-algorithms-protocols-schannel`

LIST OF ABBREVIATIONS

2FA	2-Factor Authentication
AC	Archive Center
ADMS	Administration server
AJP	Apache JServ Protocol
API	Application Programming Interface
AS	Archive Server
CAPEC	Common Attack Pattern Enumeration and Classification
CAS	Content-addressable Storage
CS	Content Server
CT	Certificate Transparency
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
CWE	Common Weakness Enumeration
CWSS	Common Weakness Scoring System
DB	Database
DBMS	Database Management System
DMS	Document Management System
DS	Document Service
EBS	Electronic Business Service
ECM	Enterprise Content Management
GUI	Graphical User Interface
HSTS	HTTP Strict Transport Security
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IAM	Identity & Access Management
IIS	Internet Information Services
IP	Internet Protocol
ISO	International Standards Organization
ITIL	Information Technology Infrastructure Library
IWA	Integrated Windows Authentication
JDBC	Java Database Connectivity
JMX	Java Management Extensions
JVM	Java Virtual Machine
KB	Knowledge Base
LDAP	Lightweight Directory Access Protocol
LDAPS	LDAP over SSL
MD	Message Digest
MIME	Multipurpose Internet Mail Extensions
MITM	Man-In-The-Middle

NAS	Network Attached Storage
OTDS	OpenText Directory Services
OVM	OScript Virtual Machine
RAM	Random Access Memory
RDBMS	Relational DataBase Management Systems
RDMS	Relational Database Management System
SAN	Storage Access Networking
SCT	Signed Certificate Timestamp
SSL	Secure Sockets Layer
SSO	Single Sign-On
STORM	Storage Manager
TLS	Transport Layer Security
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Location
XML	Extensible Markup Language
XSS	Cross-Site-Scripting

LIST OF FIGURES

LIST OF APPENDICES

APPENDIX A I. APACHE TOMCAT 8.5.X VULNERABILITIES

The complete list of Apache Tomcat 8.5.x vulnerabilities presented in this thesis may be found in a supplemental Excel file named Tomcat-vulnerabilities.xlsx.

APPENDIX A II. EXAMPLE OF INDEX.JSP

```
<%@ page session="false" pageEncoding="UTF-8" contentType="text/html;
charset=UTF-8" %>
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <title>It runs!</title>
        <link href="favicon32.png" rel="icon" sizes="32x32"  />
        <link href="favicon192.png" rel="icon" sizes="192x192" />
<link rel="apple-touch-icon-precomposed" href="favicon180.png" />
<meta name="msapplication-TileImage" content="favicon270.png" />
    </head>
    <body>
        <h2>If you’re seeing this, it runs!</h2>
    </body>
</html>
```

APPENDIX A III. WINDOWS REGISTRY – ENCRYPTION

The Windows Registry settings to disable weak transport protocol and cipher suits as presented in this thesis may be found in a supplemental text file named IIS-TLSandCiphers.txt.