Global Information Assurance Certification Paper

Tagging Data to Prevent Data Leakage (Forming Content Repositories)

GIAC (GCIA) Gold Certification

Author: Michael H. Matthee, michael.h.matthee@protonmail.com
Advisor: Richard Carbone
Accepted: April 30, 2016

Abstract

Over 2.5 quintillion bytes of data is produced in a single day. Monitoring this data is no longer effective, nor is it sustainable, using the solutions of the past. The data leakage prevention (DLP) tools of today struggle to filter through large quantities of unstructured data: images, videos, audio and the like. At the same time, these tools need to stay abreast with new sharing mechanisms, such as Facebook, Google Plus, Twitter and WhatsApp. To solve this problem, some DLP solutions are pursuing advancements in artificial intelligence. Others are beginning to collaborate with existing tools in the enterprise, like the Firewall or anti-virus product. Either way, the unfettered sharing of sensitive information from one internal department to another, is still easy to come by. A central control mechanism is needed that can determine how, where and when the information is shared from one person or group to another. By leveraging Content Management Systems, pockets of data are stored to curb the flow of information within - and out of - the enterprise. Rather than monitor every communication mechanism a workstation may have, the flow of information is constrained to a single interface - from the outset. The Content Management System becomes the portal through which data is moved from one workstation to another, starting from one division to an entire enterprise.

1. Introduction

In order to protect sensitive data, it must be secured at rest, during transit and when in use (Aaron, 2013). Vendors of data leakage prevention (DLP) solutions attempt to secure all three of these areas (Ernst and Young, 2011). For a vendor to improve the accuracy of its DLP solution, it may collaborate and integrate with other security tools within the enterprise: firewalls, intrusion detection systems or anti-virus products to name a few (Radwan & Yousef, 2014).

Current DLP solutions find it difficult to detect obfuscated data leakage (Radwan & Yousef, 2014). Hiding sensitive data within other, less noticeable files, such as images, otherwise known as steganography, is one way of leaking data covertly (Cole & Ring, 2006). Steganography is often used by malicious insiders to leak data to some outsider via e-mail (Cole & Ring, 2006). The act of malicious insiders is a growing concern within the industry (Cole & Ring, 2006). In these cases, authorised access to the information is already granted. This permits an insider to circumvent conventional detection mechanisms such as a firewall or an intrusion detection system. Together with the use of steganography, the insider is well positioned to leak the information out covertly; the leakage is obscured by a plentitude of other, seemingly normal, network traffic. Such cases can have grave consequences to the competitiveness and perception of an organization (The Washington Post, 2013). The Edward Snowden incident and the damage that it has done to the reputation of the United-States government is just one example (The Washington Post, 2013).

Sensitive information that can be stolen includes intellectual property, financial information, patient information or credit card data. A lot of progress has been made to secure sensitive information over the years. Today multiple regulations are in force to encourage and protect sensitive information. The Health Insurance Portability and Accountability Act (HIPAA) deter fraud and the abuse of patient information within the health care industry (McEvoy & Wilson, 2012). The Payment Card Industry Data Security Standard (PCI-DSS) attempts to improve trust within the conduct of daily financial transactions (Williams, 2015). Not only are there regulatory requirements, but large enterprises are aware of the risk that sensitive information leaks carry to their business.

According to a survey by Frenkel (2015), amongst 153 senior IT, information security and line-of-business professionals surveyed, 75% expressed high to very-high concern about the risk of data leakage. Of those, 84% had moderate to no confidence in their ability to secure confidential files. In a similar study conducted by Vormetric (2015), the concern is not awareness; 93% of organizations within the United States feel vulnerable to insider attacks because their mitigation measures are ineffective. What makes organizations feel this way?

According to SelectHub (2015) the top three ranking DLP products on the market are RSA, McAfee (also known as Intel Security) and Symantec. Do these DLP solutions reassure confidence in securing confidential files within the enterprise? To answer that question, a Gartner report by Reed & Wynne (2016) is used to identify important focus areas that are lacking amongst DLP solutions (see Table 1).

Table 1: Areas where current DLP solutions lack when protecting sensitive data. Information sourced from a Gartner report by Reed & Wynne (2016).

Focus Area	Objective	RSA	McAfee	Symantec
Platform Independence	The tool is able to run on all platforms, computer architectures and operating systems.	☒	☒	☒
Protecting information in the cloud	The tool is able to protect corporate information that is stored online in the cloud.	☑	☒	☒
Data classification	The tool can integrate with other data classification and categorization tools.	☒	☑	☒

The product from Symantec has limited support for the Macintosh platform and no support for Linux operating systems (Reed & Wynne, 2016). When it comes to protecting information online, in the cloud, the product only integrates natively with two technologies: Box and SharePoint Online (Reed & Wynne, 2016). Information on its support for third party data classification tools could not be found.

In 2010, Intel acquired McAfee. Consequently, their DLP product now falls under the Intel Security technology suite (Reed & Wynne, 2016). The McAfee product does not

integrate natively to cloud storage providers (Reed & Wynne, 2016). There is limited support for the Macintosh platform and no support for Linux operating systems (Reed & Wynne, 2016). The McAfee product integrates with third party data classification tools, including Titus and Boldon James (Reed & Wynne, 2016).

Although their DLP tool is still considered to be a major player (SelectHub, 2015), RSA decided to exit the DLP product business in 2015 (Reed & Wynne, 2016). The tool is primarily geared for Microsoft Windows platforms. It is arguably one of the first DLP products to integrate with cloud storage (Wah, 2016).

The best DLP solutions in the marketplace lack in the areas of encryption, access control and controlling data flow over social communication tools – whether internal or public (Shabtai, Elovici, & Rokach, 2012). They are not pre-configured to monitor custom proprietary protocols, which are unknown to the community at large.

Further to this, current DLP solutions lack in their ability to protect large amounts of unstructured data, i.e. various types of intellectual property like source code, customer lists, and product designs (Shabtai, Elovici, & Rokach, 2012). DLP solutions also struggle to enforce data leakage prevention to other internal departments, business units or small teams within the same organisation (Shabtai, Elovici, & Rokach, 2012). Sensitive information can be dispersed from a more restrictive business unit to a less restrictive one; easing the means of bypassing tight security controls that is enforced elsewhere (Cole & Ring, 2006). These are some of the concerns that current DLP solutions face.

2. Current Solutions

In order to mitigate these gaps, an organisation might employ one or more categories of security measures together in order to defend against data leakage. Supporting the notion of a defense-in-depth strategy, a holistic approach to protecting data within the enterprise is recommended (Ernst and Young, 2011). Four categories of security measures achieve this purpose.

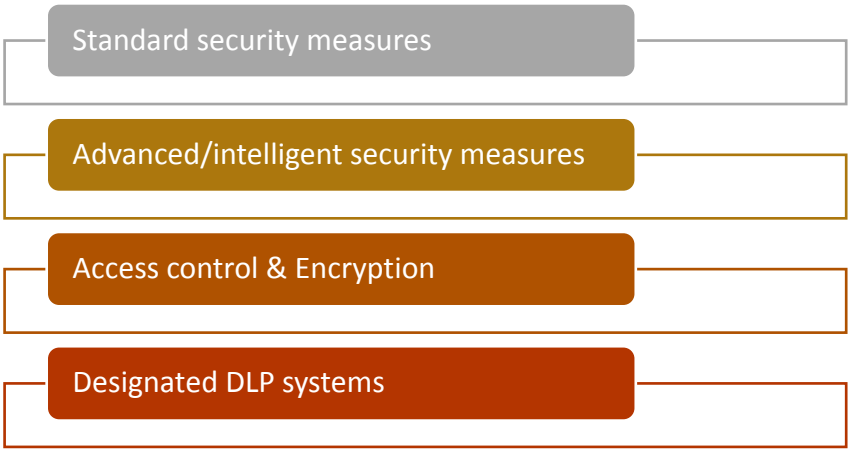

Figure 1: Various categories of security mechanisms that help to prevent data leakage. Image was re-created from Ernst and Young (2011).

Starting with standard security measures, these include: firewalls, intrusion detection systems and anti-virus software. These mechanisms are not sufficient to deter the resolve of a malicious insider alone; they primarily focus on mitigating external threat actors instead (Shabtai, Elovici, & Rokach, 2012). However, they remain invaluable in forming a defense-in-depth data protection strategy.

The advanced/intelligent security mechanisms employ behaviour analysis routines such as machine learning to detect unusual user activity (e.g. based on keystrokes and mouse patterns), abnormal data access routines or suspicious email exchange patterns (Shabtai, Elovici, & Rokach, 2012).

Device control, access control and encryption are basic hygiene measures to protect against unauthorized access to data. These mechanisms are vital to the safety of an organization, well beyond the confines of data leakage prevention alone (Ernst and Young, 2011).

Designated DLP products can take many forms and fall into – one or more – of the following categories (Shabtai, Elovici, & Rokach, 2012):

- Context-based solutions inspect the contextual information of monitored data, such as the route of transfer, source, destination, file type or time stamps. These solutions collaborate with other security mechanisms within the enterprise, such as firewalls or anti-virus, in order to obtain a holistic view of data flow within the

enterprise. For example, the DLP solution may collaborate with a firewall to decide whether data can be shared, or not.

- Content-based solutions inspect the actual data that is transmitted across the network. Observation techniques include the use of keywords, e.g. detecting a social security number, or natural language linguistics, e.g. analysing the frequency at which key terms occur.
- In content tagging, the consumer declares sensitive data by providing customized rules to the DLP solution. These tagging solutions require that the consumer is well organized and is able to identify all sensitive data within an enterprise. The effectiveness of this solution depends on the conscientious efforts of the customer, and not to overlook any sample of data/information that may be critical.

Historically, the content-based approach was favored - data leaks would be detected by looking for keywords, using regular expressions or hash values against the content (Nikitinsky, Sokolova, & Pshehotskaya, 2014). This worked well for analysing structured data, but with the ease of access to email and social networks today, confidential information can be rephrased or reformatted to avoid detection through these channels. Attempts have been made to improve on this approach nonetheless. The introduction of machine-learning techniques has enabled DLP solutions to perform more advanced analyses; the detection of photographed credit cards being one example (Nikitinsky, Sokolova, & Pshehotskaya, 2014). Yet, future trends favor the inclusion of a content tagging strategy due to the limitation of keyword lists (Nikitinsky, Sokolova, & Pshehotskaya, 2014).

The traditional content-tagging solutions employ agents that scan the target workstation or server, detect any stored data that violates its data distribution policy, and thereafter takes action on it. This could be to quarantine, encrypt or relocate the data (Shabtai, Elovici, & Rokach, 2012). One drawback of this approach, however, is that action is only taken after the data has been distributed to the workstation or server. Greater emphasis can be placed on preventing the unwieldy proliferation of data before it occurs. Another drawback is that only the portions of data that was pre-configured as sensitive are eventually tagged. This particularly holds true for unstructured data – information that is formed over say social media, email or video presentations. This form

of data sharing is malliable, dynamic and difficult to direct. Sensitive information can easily bypass keyword-based content matching routines when transferred in such an unstructured form (Shabtai, Elovici, & Rokach, 2012).

In the 2 years leading up to 2012, more than 90% of the world's data was generated at the time. Eighty percent of this data was unstructured (IBM, 2012). The enterprise is no different, and with such a flood of unstructured data amongst her employees, it is difficult to detect the leakage of sensitive information amongst so much noise. And yet, missing a data leak can be detrimental. In the followup to August of 2007, an employee at the Nuclear Laboratory in Los Almos transmitted several emails of confidential information, posing a serious threat to the national security of the United States (Shabtai, Elovici, & Rokach, 2012). By simply rephrasing or formatting unstructured information, it can bypass leakage detection (Stephenson, 2014); a dependable content classification and tagging component is essential to DLP solutions of the future (Nikitinsky, Sokolova, & Pshehotskaya, 2014).

As mentioned in the preceding paragraphs, current DLP solutions have a number of gaps; improvements can be made. A completely free and open-source data tagging DLP solution is suggested that addresses many of these concerns.

3. Open-source Data Tagging System

The proposed data tagging system must be able to sequence, tag and detect data leakage across all forms of structured and unstructured data within the enterprise. Now to compliment that, the envisioned system must also be economical and readily customizable through the use of freely available and open-source software components.

Three components form the backbone of this data tagging solution. The first component must collect and classify organizational data at rest. The Java Content Repository (JCR) is a free and open-source technology that already performs this task well. Secondly, a custom-build data leakage preventions service (DLP service) is needed that demarcates the proliferation of “data in motion” within the enterprise and abroad. It must act as the gateway for data both entering and leaving the user’s workstation. This includes data moving over e-mail, Instant Messaging or document file sharing. The final component must secure data in use by redacting or sanitizing sensitive data, monitor its

distribution and determine how, where and when it may be shared. A Content Management System (CMS) serves as a centralized control mechanism on how the organization's collective information is used. Magnolia (2016) is a free and open-source CMS that satisfies this final requirement.

The hypothesis is that together these components could increase both the accuracy, as well as the performance of data leakage prevention in the enterprise. By providing a readily extendable and customizable framework for DLP, one can cover all of the network protocols and data artifacts within the enterprise – even if custom protocols are in use. Finally, the suggested data-tagging mechanism requires minimal human intervention and setup costs, which would imply fewer burdens on the enterprise.

3.1. The Java Content Repository

One remarkable difference between the Java Content Repository and other forms of data-storage mechanisms is that it can make associations amongst data artifacts. A data artifact can range from anything visual like a PNG image stored on a file-system to a developer that performs a code commit to a GIT version control system. The image stored on the file-system is a stationary sample of information – data at rest. Likewise, code committed over the GIT version control system is an example of data in transit. JCR acts like a unified interface to all of this data - whether stationary or in-transit across a network segment. It extracts, assembles and aggregates diverse forms of pre-existing data through its collection of sequencer and connector components. Not all of the enterprise data that JCR extracts is accessible via a simple query to a file-system. Important data could be stored within system applications like SharePoint, OneNote or Active Directory. Databases like Oracle, Postgress or MySQL could also store corporate information that must be tagged and classified. JCR connectors can establish connections to a large array of disparate data repositories and content storage mechanisms deployed throughout the enterprise. This allows content from across the enterprise to be indexed and tagged for use – however and wherever it may be generated from.

The sequencers identify the type of data that is involved – it could be an image, audio file, database content, office document or most other forms of data. After detection, it cracks open each data artifact and sequences it into a JCR defined hierarchical format.

By doing so, the system is able to inspect the insides of data artifacts, compute hash values of each constituent part and tag it categorically.

Architecturally, both the sequencers along with the connectors form the backbone of a JCR interface as depicted in Figure 2 below.

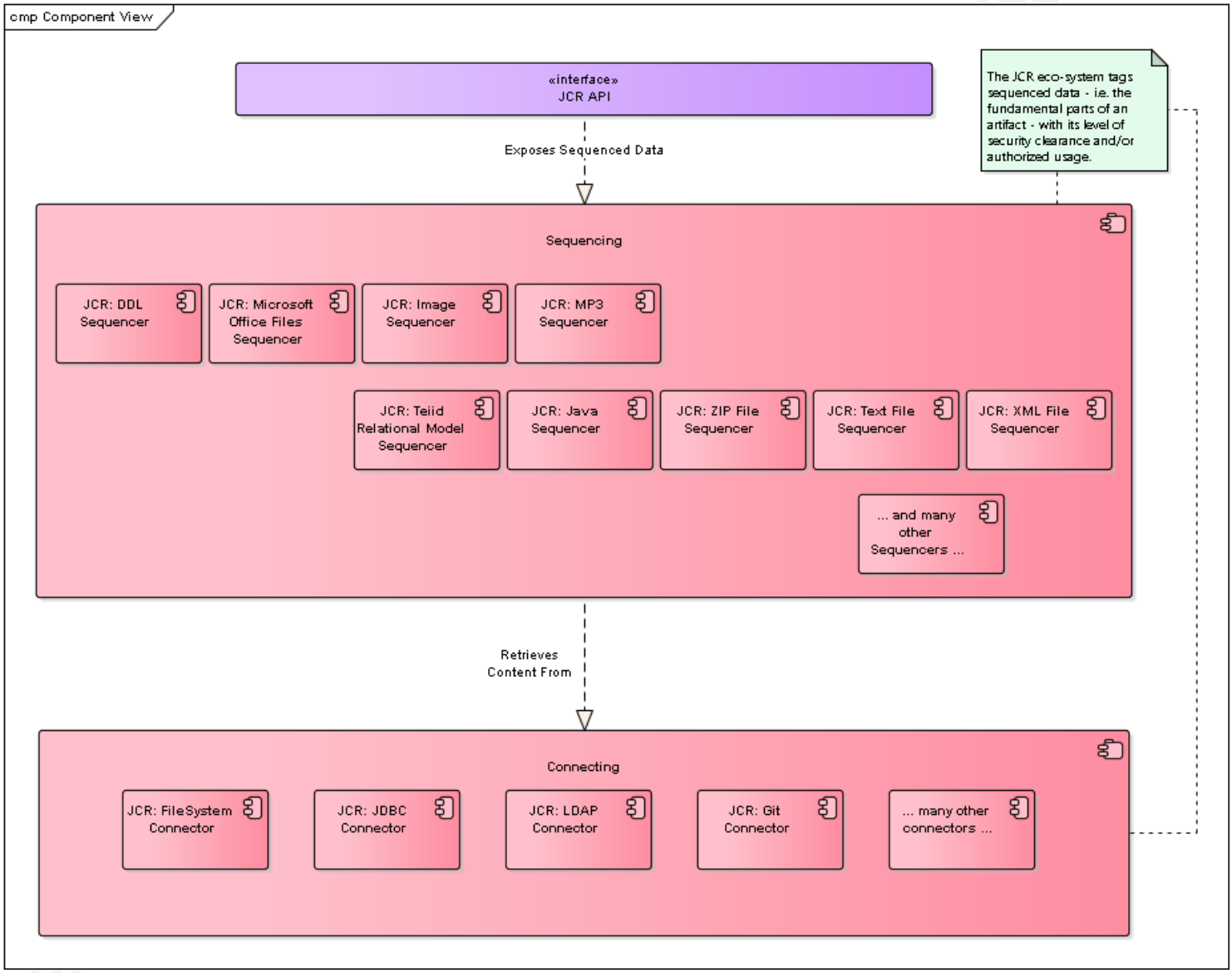

Figure 2: The components of JCR that aggregate, sequence and index corporate data. Image was adapted from (jboss, 2013).

The JCR application programmer interface is used to access and store sequenced data (indicated as the purple block within Figure 2). Sequenced data refers to information that has been broken down into its constituent parts. Various sequencers exist to perform this decomposition process (indicated as the first top red block of Figure 2). Take an MP3 file as an example. The MP3 sequencer extracts ID3 metadata from the MP3 file.

This includes the track: title, author, album name, year and any additional comments. In turn, this meta-information helps to tag and classify the MP3 file contents correctly.

What do we gain from performing this JCR indexing exercise? Suppose that a malicious user hides a top-secret Excel spreadsheet within a public Word document and attempts to share it on the internet. The Word document is in public domain – i.e. users may distribute it freely, to whomever and wherever they choose. However, the Excel spreadsheet is very sensitive. Firstly, the JCR sequencing process views the amalgamated file – the spreadsheet hidden within the Word document – as two distinct files. Because of this, the JCR data tagging system is impervious to this particular data leakage technique. Secondly, the JCR system cross-references the embedded Excel spreadsheet within its repository index. Upon doing so, it reveals that the Excel sheet is top secret and may not be shared publicly; an attempted data leak is thwarted.

Inspecting the inner bits and bytes of binary content for hidden content is somewhat harder. By default, the JCR system does not perform analysis on binary blobs such as images or video streams. However, configuring the JCR repository to use MongoDB (MongoDB, 2016) as its back-end processing data-store, this becomes possible. MongoDB is a high performance data-store that enables image processing and computer vision libraries to run against any binary data. One such library is OpenCV (OpenCV.org, 2016) and was used in this proof of concept to interrogate the information that is portrayed in binary images.

3.1.1. The repository's hierarchical data format

A JCR repository stores, correlates and associates data that is hierarchical in nature, much better than relational SQL databases can (Jboss, 2013). However, not all data has a hierarchical structure. Take a folder containing a collection of arbitrary images as an example; it does not necessarily relate to anything else and yet it can consume a lot of space. Content repositories like JCR do not perform well in handling large amounts of flat, non-hierarchical data (Jboss, 2013). While this may hold true for the data itself, on a meta-data level the situation looks different. On a meta-level, one can structure and organize that folder of arbitrary images in numerous ways. One can categorize them according to the creator of each photo, the location of the shoot, its creation date; or one could attach some intelligent labelling, subject and descriptive text to it. This hierarchical

meta-structure allows you to give data new meaning, to make sense of it in context of its history and current setting. From an information security perspective, these meta-data characteristics are particularly useful towards forming a taxonomy of data access rights or privileges. By tagging data according to this taxonomy of rights and its intended audience, one can define where/how the information sharing occurs.

Governments and corporate institutions classify data and information according to some high-level criteria. These classification categories define the data's permitted use and distribution. Figure 3 is an example of a classification tree used by the British government. At the one extreme, there is public information – content one may wish to share with the rest of the world – e.g. marketing material. On the other extreme, one may have very sensitive information e.g. the operational manual for a nuclear bomb – this is content one would probably want to label as top secret.

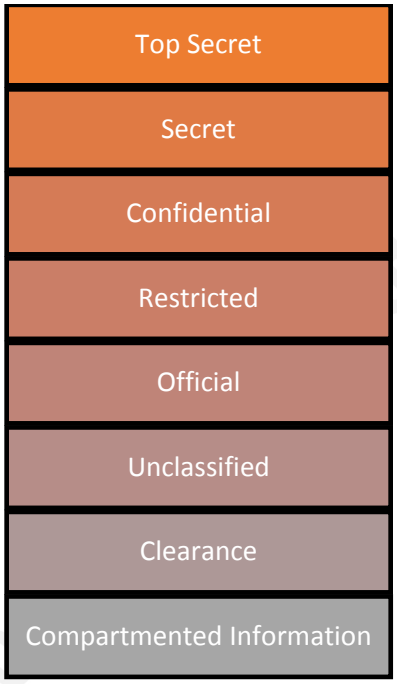

Figure 3: British classification levels for data. Image was adapted from (Fowler, 2003).

There is no need to limit the categories and the classification of information to those listed in Figure 3. Rather tailor data classification categories according to the unique circumstances of one organization (Fowler, 2003). Also, consider the impact that such classification categories can have upon the efficacy and performance of the

enterprise as a whole. An incorrect, or out-of-date, classification of information can have severe cost implications to an organization (Bergström & Åhlfeldt, 2014).

Besides a classification tree, the JCR repository maintains a built-in version control system. Any incremental updates to an artifact of data is automatically re-sequenced and versioned (Jboss, 2013). This provides snapshots of the data as it evolves over time – making differential comparisons or rollbacks on data possible. In doing so, the classification of a data artifact can be revised with greater ease, by analyzing delta changes only.

An example is in order. Presume a text snippet is stored on a local file system. How will JCR process it? Firstly, the JCR file system connector will pick up the file from the file-system and index it. After the indexing process is complete, the text sequencer breaks up the text document to form a JCR node graph representation. Assume that the file contains the text from Figure 4.

```
# This line should be ignored completely
text371449635398431populated
rudimentary data here<36438936438936>
nothing to HIDE-5500005555555559-
# So should this one
000-10-0405
 4444444444444448
```

Figure 4: Sample text to be sequenced by JCR.

There are a number of ways that JCR can sequence this content into its constituent parts. It does not necessarily need to sequence text into words. The sample text can be broken up according to some delimiter, e.g. a comma or a space, a column index or a regular expression; or by means of file position or index. A configuration file can also specify some text patterns (or words) to ignore while constructing the JCR node graph equivalent. This feature is useful to ignore non-sensitive, ancillary data from overloading the data monitoring mechanism. For example, one could safely ignore numerous tags, delimiters, fonts and formatting options, as they are seldom unique and sensitive to an organization. However, referring to the sample text of Figure 4 again, it holds some personal identifiable information (PII). In this case, a text sequencer based on regular expressions will correctly identify the sensitive content. The regular expression listed in Figure 5 identify can some common credit card numbers.

```
^(?:4[0-9]{12}(?:[0-9]{3})?              # Visa
 | 5[1-5][0-9]{14}                      # MasterCard
 | 3[47][0-9]{13}                       # American Express
 | 3(?:0[0-5]|[68][0-9])[0-9]{11}       # Diners Club
 | 6(?:011|5[0-9]{2})[0-9]{12}          # Discover
 | (?:2131|1800|35\d{3})\d{11}          # JCB
)$
```

Figure 5: Regular expression for identifying credit card numbers within text.

A sequencing process produces a corresponding JCR node graph of the input text. In this case, running the JCR text sequencer – with the regular expression of Figure 5 and the sample text of Figure 4 – produces the node graph shown in Figure 6 below.

```
fixedWidthFile.txt jcr:primaryType=nt:unstructured jcr:mixinTypes=[mode:derived]
        -mode:derivedFrom=/files/fixedWidthFile.txt/jcr:content
        text:row jcr:primaryType=nt:unstructured
                text:column jcr:primaryType=nt:unstructured jcr:mixinTypes=[text:column]
                        -text:data=371449635398431
        text:row[2] jcr:primaryType=nt:unstructured
                text:column jcr:primaryType=nt:unstructured
```

Figure 6: Detecting and sequencing sensitive data using JCR.

On the top line of the output, a JCR file node represents the sample text file as an artifact. Every subsequent node is associated with a row of text within the file. Depending upon the text sequencer's configuration settings, other data pieces may also be listed within the output graph. This may include a timestamp of the file's last modification date, the author or the file's purpose – e.g. a title, subject or chapter headings.

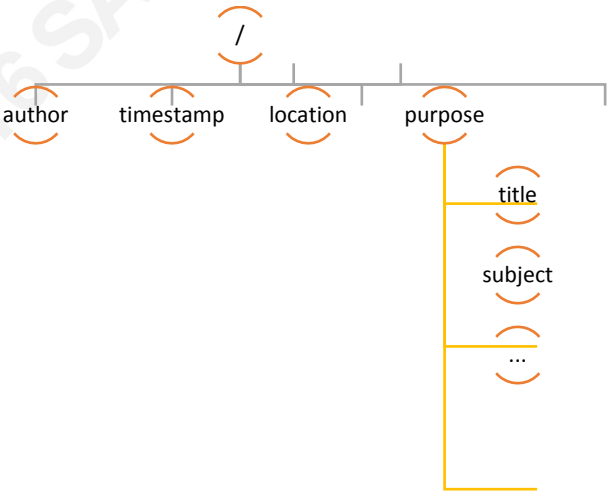

Figure 7: Structuring the meta-data of a data artifact – an example.

The hierarchical structure of the JCR repository is well suited for the organization and the classification of data. However, what is the recommended way to perform and structure such classifications?

3.1.2. Classification of existing Data

Before attempting to classify data, information policies must be in place for the enterprise. These policies prepare the organization from a legal perspective and ensures that all of the enterprise data is analyzed and classified appropriately (Peter, 2015). Data items such as social security numbers, credit card information and personal health information is easy to detect and classify accordingly – well-defined regulations exist for these. Classification becomes tricky when including organizational specific requirements such as intellectual property.

For many organizations, the classification of data remains an onerous process, employees are enlisted to tag every data artifact manually – for both legacy as well as newly created data (Stephenson, 2014). Automated data classification tools are emerging but they are not yet in general use. One likely reason for the low adoption of these tools – i.e. to classify data in bulk – is that their user-friendly interfaces and interoperability features with existing DLP tools, are still maturing (Stephenson, 2014).

However, using the JCR system, data is classified using automatic data classification scripts. This reduces the manual intervention or effort required by personnel.

3.1.3. Forming the classification tree

It is best to drive data classification according to a policy (Peter, 2015). One major benefit on using JCR is that it can classify pre-existing data. To demonstrate this, an existing standalone JCR repository from the Apache Jackrabbit project is used (Apache Foundation, 2016).

Download the *jackrabbit-standalone-2.12.0.jar* file from (Apache Foundation, 2016) and start it from the command line. After starting up, the standalone project has a demo-landing page available on the localhost, port 8080.

Figure 8: Automatic data classification example: uploading of data artifacts.

The demo-landing page (shown in Figure 8) has an upload facility to populate the JCR repository with sample data from the Internet. Navigate to the *Populate* link under the *Default workspace* section and populate the repository with some data. Note that this is not the only way to populate the JCR repository. The proof of concept code, mentioned later, does not use this upload interface. Instead, it sequences all the artifacts from the file-system of a client workstation by default.

Next, define a *mixin* property in the JCR configuration that will hold the classification category for uploaded data. To do this, create a new JCR node type with the declaration listing of Figure 9.

```
// The namespace declaration
<ns = 'http://namespace.com/ns'>

// The namespace of our custom data tagging property
<ex = 'http://example.com/jcr/cnd'>

// Node type name
[ns:DataTagging]
```

Figure 9: Declaring the classification mixin node attribute.

This declaration defines a new node type called *DataTagging* in the JCR system. A node type is a meta-value associated to every data artifact or constituent of a data artifact within the JCR eco-system. The rest of the node configuration defines a property called *ex:classification*. The value of this property determines how the respective data artifact may be distributed: e.g., publicly, only within the enterprise or only amongst

certain people. The configuration listing below happens to have a default setting of *public*. This means that if no classification scripts are run to alter this value, then the data may be shared with everyone.

```
// This node type supports orderable child nodes
orderable

// This is a mixin node type
mixin

// Nodes of this node type have a property called 'ex:classification' of type
STRING
- ex:classification (string)

// The default values for this
// (multi-value) property are...
= 'public'

// This property is the primary item
primary

// and it is...
mandatory autocreated protected

// and multi-valued
multiple

// It has an on-parent-version setting of ...
version
```

Figure 10: Adding a JCR classification tag to data artifacts.

After registering the *mixin* type to the JCR instance, verify that it is active. Execute a WebDav query (similar to the one listed in Figure 11) against the uploaded content. JCR should respond with an *ex:classification* value of *public*, which is the default value configured in this example.

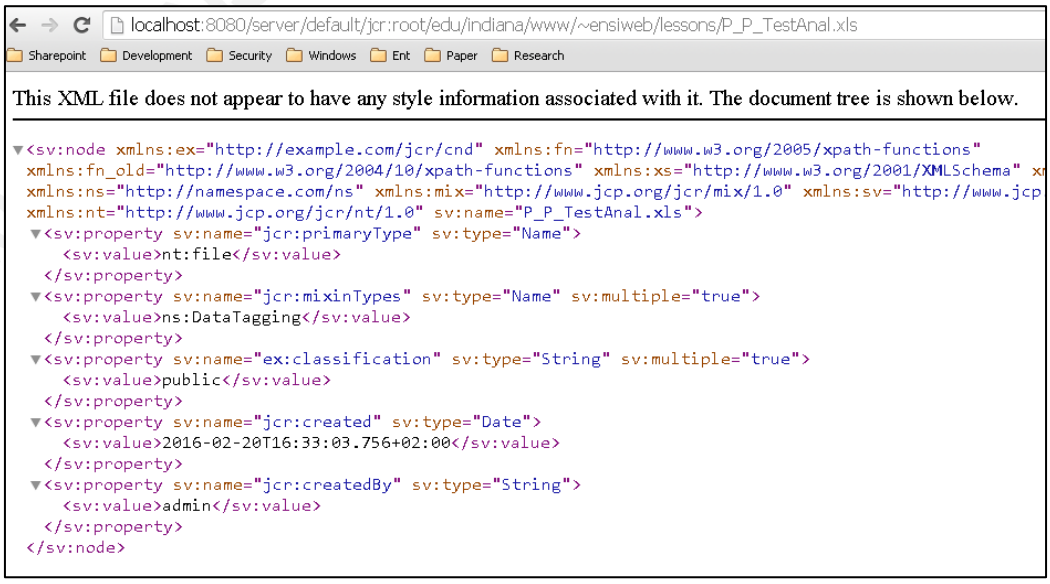

Figure 11: Automatic data classification example: retrieving the classification category from JCR.

The result of this query determines how the data (in this case a financial analysis spreadsheet named *P_P_TestAnal.xls*) is classified. By inspecting this classification value, the tagging system knows how and to whom it may share the file.

3.2. Constructing the data leakage prevention service

Data leakage occurs either through digital or physical means. Digital avenues include e-mail, web mail, logs, wikis, making physical back-ups using removable media and forums - anything that connects over a network protocol. Physical avenues for data leakage include taking photos using a camera, posting hard copies of paper-printed material, examining portable devices such as a mobile phones or laptops, shoulder watching, social engineering and physical theft.

A workstation has many channels through which data can be distributed and shared. The purpose of the data leakage prevention service (DLP service) is to prevent the unfettered leakage of sensitive data. One could decide to monitor all of these distribution channels – USB, disc drive, web sites, e-mail and so on - however, a simpler solution is to constrain communications to a single interface instead.

An enterprise content management system (CMS) is a technology geared to store the content of an enterprise. Employees use a CMS to search, store, retrieve or share information with one another. In this suggested data tagging solution, nobody is allowed to distribute data directly: e.g. via e-mail, network protocols, CD drives or USB devices. Instead, workstations must connect and cross-reference the JCR interface of a CMS, in order to upload, share and distribute information. Data queries to the JCR interface return with a tag value – indicating the class of data, its purpose and where/how it may be distributed. The DLP service intercepts the transfer of information and refers to the associated tag value, in order to enforce data distribution policies. Depending on the rules, the distribution of information may be sanctioned or not.

Here is an example: assume that a user wants to share and distribute a spreadsheet named ‘FinancialSecrets.xls’ across the organization via e-mail. Since distribution channels are not directly accessible from the workstation, the user navigates over to the content management system. The CMS offers the user an e-mail facility to share the information. Extending the CMS with communication channels, such as the e-mail

facility in this case, is made available by a DLP service (Matthee, 2016). The example below utilized protonmail – a free e-mail software client – to e-mail the information.

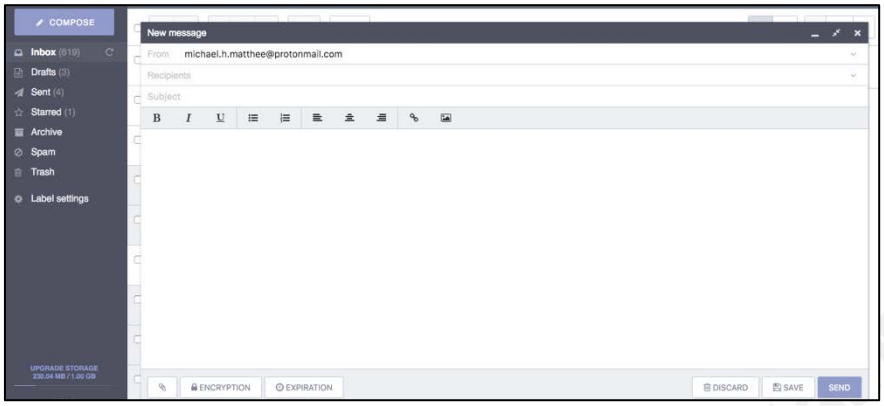

Figure 12: Protonmail e-mail services hosted by an enterprise content management system.

The attachment facility of the e-mail service is restricted; it can only attach documents that are available from the CMS. This means that the Excel spreadsheet is already analyzed, classified and tagged against a distribution policy. Any text written inside the posted e-mail is also sequenced by the JCR system before it is sent onto the DLP service for delivery to the destination. The mechanism for sequencing and classifying the text is based upon the example described in Section 3.1.3.

The DLP service component is built on top of Spring Integration (Spring, 2016); a framework that is geared to integrate with any service in industry. Spring Integration supports several service types outright, including twitter, e-mail, FTP, web protocols and so on. If a service is not catered for, it can be accomplished by plugging in a custom protocol adapter. The available integration mechanisms for the DLP proof of concept code is shown in Figure 13.

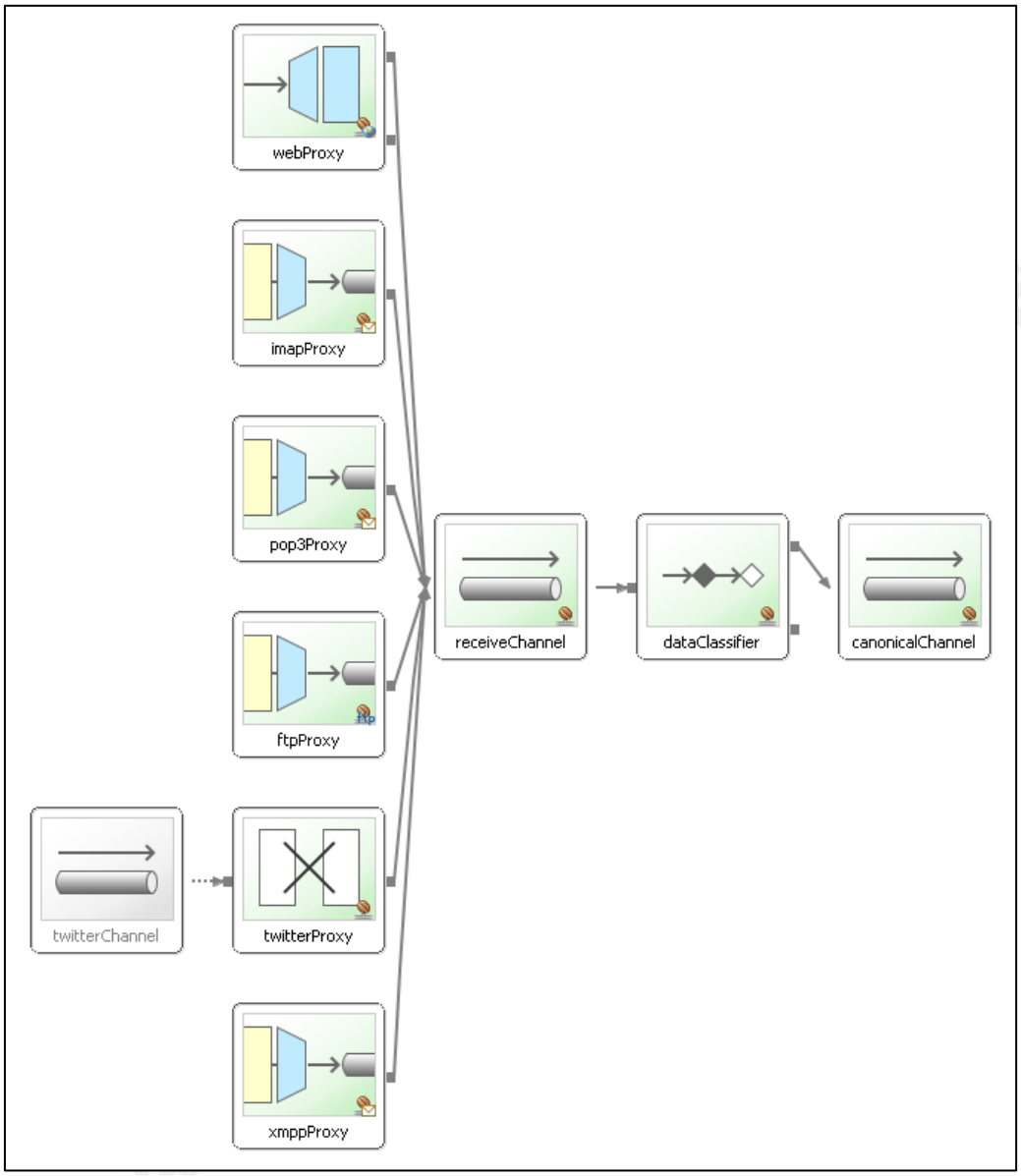

Figure 13: Data leakage prevention service: integration components

The DLP service intercepts multiple transport types including FTP, twitter and XMPP. For incoming/outgoing e-mail, it listens on imap and pop3 endpoints, before sending it on to the data classifier. The classifier sequences the e-mail message – the written text along with the attachment – and determines if the e-mail may be shared to the destination address or not. The code for the DLP service is freely available on GitHub (Matthee, 2016).

3.3. Integrating to a Content Management System

The final component of the data tagging system secures data in use. This is done by employing a Content Management System (CMS). A CMS provides a central control mechanism to redact or sanitize sensitive data, monitor its distribution and determine how, where and when it is shared. Magnolia (2016) is a free and open-source CMS that fulfills this requirement and is used to demonstrate the data tagging solution.

The Magnolia CMS contains an embedded JCR instance. This eases the integration of Magnolia with the other two components, namely: the DLP service and the JCR data classification schema.

The Magnolia CMS is deployed to a web container. Magnolia consists of two web-applications, namely: *magnoliaAuthor* for configuring the CMS, and *magnoliaPublic* that serves as the entry point for end-users.

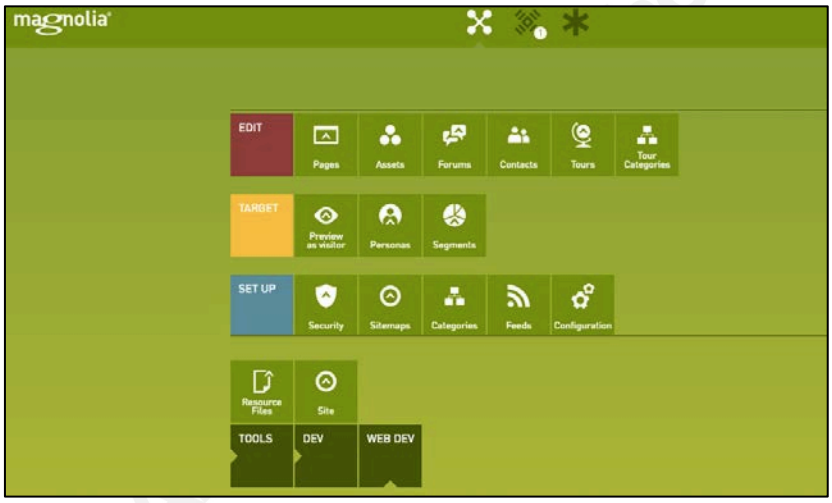

Figure 14: The landing page of magnolia (Author). It is used to integrate and configure the CMS for the purposes of data tagging.

The embedded JCR instance is configured under the tools menu of the *magnoliaAuthor* landing page. The data classification *mixin* of Section 3.1.3 is defined in this configuration page. After configuring Magnolia with this *mixin*, a Groovy script populates the *ex:classification* property of each data artifact in the CMS with a classification value – whether it may be a document, text page or any other data artifact. The Groovy script from Figure 15 performs such an exercise by setting data node values.

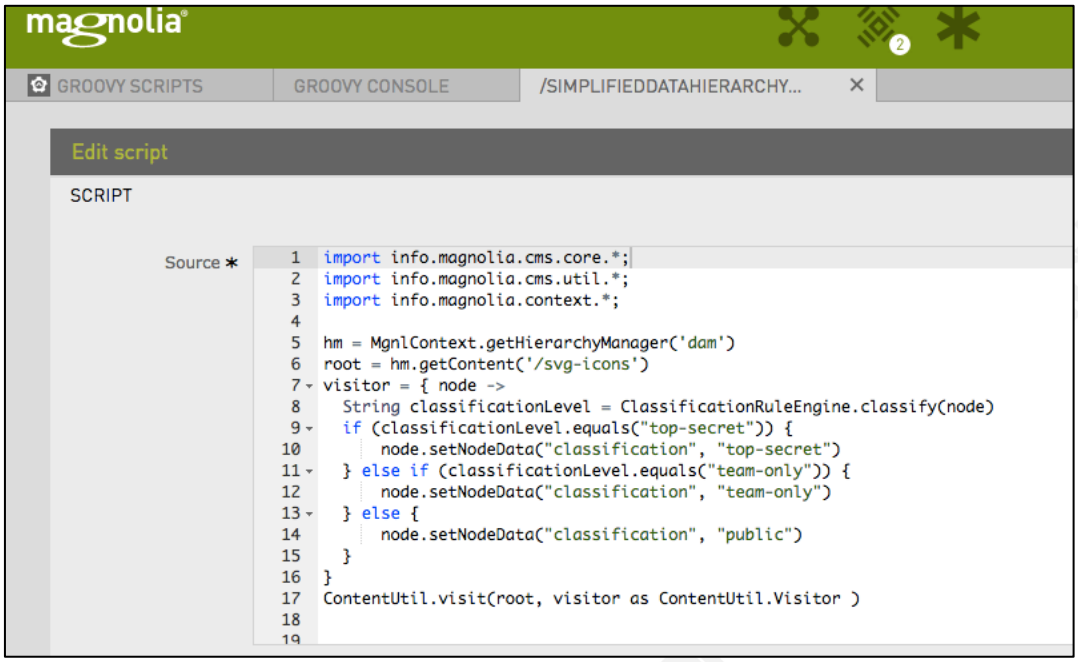

Figure 15: Adding the data classification script to Magnolia CMS.

In this script, the *ClassificationRuleEngine* contains a number of rules for detecting intellectual property, health or other personal information. The *classification* attribute of each data node is then assigned the computed value of this function.

A security filter is then added to Magnolia to enforce the data classification rules. The security filter wraps around the build-in *SiteUriSecurityFilter* to either sanction or prohibit a user from accessing a resource.

```
public class DLPAccessControlFilter extends SiteUriSecurityFilter {

        public static Logger log = LoggerFactory.getLogger(DLPAccessControlFilter.class);

        @Override
        protected boolean isAuthorized(HttpServletRequest request) {

                try {
                        // Perform JCR lookup
                        log.info("Performing JCR lookup");
                        HierarchyManager hm = MgnlContext.getHierarchyManager("website");
                        Content node = hm.getContent("/demo-project");

                        // Determine data classification level
                        log.info("Determining classification level");
                        String classification = node.getNodeData("classification").getString();

                        // Perform access control
                        log.info("Performing access control");
                        return isUserSanctionedToAccess(classification);
```

Figure 16: Controlling access to resources using data classification tags.

4. Conclusion

In summary, confining the distribution of information via a CMS alone is possible. Most of the functionality for work – such as e-mail and Instant Messaging – can still be performed. Work artifacts such as documents (MS Word, PDF, MS Excel, images etc.) or software (source code, configuration files or tools) is shared and distributed amongst employees. There is no reason why it may not be stored centrally before it is shared with others.

The benefits of forcing employees to share and distribute information through a central CMS and DLP service are:

1. Data loss prevention: Every work artifact that is produced must be checked in to the CMS before it can be shared with others. This mitigates the loss of key information and also allows other team members to readily access backed up information.
2. Tacit knowledge made explicit: Team members create and share a lot of information in an unstructured form, i.e. text messages over Instant Messaging or e-mail. By sequencing these conversations, work-related discussions can be stored explicitly for future reference.
3. Data leakage prevention: The DLP service ensures that the flow of information within and to the outside of the enterprise follows clear paths. For example, sharing a blueprint of a nuclear power plant design with a call-center agent, does not fall under the agent's designated job responsibilities. Such unsanctioned information sharing is detected and prevented by the DLP service.
4. Improved accuracy: By leveraging MongoDB as the back-end data-store of the JCR repository system, binary content can be scanned and inspected by means of image processing and computer vision techniques. This mitigates the leaking of sensitive information through covert techniques such as taking screenshots, steganography or obfuscation.
5. Inventory control: Only tools sanctioned by the enterprise may be installed to client workstations. These tools are stored centrally within the CMS for sharing and distribution to employees with granted rights. This mitigates the

use of tools for covert information leakage such as steganography, encryption or protocol tunneling.

Concerns on using this setup may include the sense of freedom amongst employees to create and share content, as and with whomever, they wish. The DLP system does not however restrict employees from accessing or contributing towards personal content such as: Facebook, private e-mails or web-browsing. It does, however, analyze the shared content to detect and prevent the leakage of any confidential information from the enterprise.

A working prototype (Matthee, 2016) is presented (Reed & Wynne, 2016) that makes use of Java Content Repositories and Content Management Systems to prevent sensitive data leakage. Content Management Systems can be assembled to act as portals – or gateways – for the distribution of information inside and external to the enterprise. In this manner, you can curb the flow of information, such that the right information reaches the right people, appropriately.

5. References

Aaron, W. (2013). *Enterprise Security: A Data-Centric Approach to Securing the Enterprise.* Birmingham, United Kingdom: Packt Publishing.

Apache Foundation. (2016, February 20). *jcr*. Retrieved from jackrabbit.apache.org: https://jackrabbit.apache.org/jcr

Bergström, E., & Åhlfeldt, R.-M. (2014). Information Classification Issues. *19'th Nordic Conference, NordSec. XII*, pp. 27-41. Switzerland: Springer.

Cole, E., & Ring, S. (2006). Insider threat: Protecting the enterprise from sabotage, spying, and theft. In E. Cole, & S. Ring, *Insider threat: Protecting the enterprise from sabotage, spying, and theft* (pp. 49-100). Rockland, Maine, United States of America: Syngress.

Colin, T. (2015, May 2). Data classification – the foundation of information security. *Network Security*, pp. 8-11.

Ernst and Young. (2011, October). *EY_Data_Loss_Prevention.* Retrieved January 2016, 2016, from www.ey.com: http://www.ey.com/Publication/vwLUAssets/EY_Data_Loss_Prevention/$FILE/EY_Data_Loss_Prevention.pdf

Fowler, S. (2003, February 28). *information-classification-who-846.* Retrieved from sans.org: https://www.sans.org/reading-room/whitepapers/auditing/information-classification-who-846

Frenkel, K. A. (2015, February 2). CIO Insight. p. 2.

Gartner, Reed, B., & Wynne, N. (2016). *Magic Quadrant for Enterprise Data Loss Prevention.* Gartner. Stamford: Gartner.

IBM. (2012). *idrevents*. Retrieved February 23, 2016, from www-304.ibm.com.

Jboss. (2013, November 10). *Why+use+a+repository*. Retrieved January 31, 2016, from docs.jboss.org/author/display/MODE40: https://docs.jboss.org/author/display/MODE40/Why+use+a+repository

Jinhyung, K., Park, C., Jun, H., & Hyung-Jong, K. (2013). Privacy Level Indicating Data Leakage Prevention System. *KSII Transactions on Internet & Information Systems*, 558-575. Retrieved from http://web.a.ebscohost.com/

Magnolia. (2016, April 27). *magnolia-cms*. Retrieved from documentation.magnolia-cms.com: https://documentation.magnolia-cms.com

Matthee, M. H. (2016, April 28). *michaelmatthee/DLPService*. Retrieved from github.com: https://github.com/michaelmatthee/DLPService

McEvoy, S., & Wilson, P. (2012). Health IT JumpStart: The Best First Step Toward an IT Career in Health Information Technology. In *HIPAA Regulations.* Indianapolis: Sybex.

MongoDB. (2016, April 22). *https://www.mongodb.org/*. Retrieved from mongodb: https://www.mongodb.org/

Nikitinsky, N., Sokolova, T., & Pshehotskaya, E. (2014). DLP Technologies: Challenges and Future Directions. *Proceedings of the International Conference on Cyber-Crime Investigation and Cyber Security* (pp. 31-36). Kuala Lumpur: Asia Pacific University of Technology and Innovation. Retrieved from https://www.researchgate.net/publication/268462340_DLP_Technologies_Challenges_and_Future_Directions

OpenCV.org. (2016, April 22). *http://opencv.org/*. Retrieved from http://opencv.org/: http://opencv.org/

Oracle. (2016, April 22). *jvisualvm*. Retrieved from http://docs.oracle.com/: http://docs.oracle.com/javase/6/docs/technotes/tools/share/jvisualvm.html

Peter, S. (2015, June 5). Data classification and DLP tools. *SC Magazine: For IT Security Professionals (15476693), 26*(6), pp. 38-39. Retrieved February 20, 2016

Radwan, T., & Yousef, S. (2014, July 30). Data Leakage/Loss Prevention Systems (DLP). *NNGT Journal: International Journal of Information Systems, 1*, 13-19.

Securosis, L. (2010, October 21). *DLP-Whitepaper.* (C. Pepper, Ed.) Retrieved January 23, 2016, from https://securosis.com: https://securosis.com/assets/library/reports/DLP-Whitepaper.pdf

SelectHub. (2015, September 26). *data-loss-prevention*. Retrieved January 23, 2016, from https://app.selecthub.com: https://app.selecthub.com/categories/data-loss-prevention

Shabtai, A., Elovici, Y., & Rokach, L. (2012). *A Survey of Data Leakage.* New York, United States of America: Springer US.

SmartBear. (2016, April 22). *smartbear.com*. Retrieved from https://smartbear.com/: https://smartbear.com/

Spring. (2016, April 22). *spring-integration*. Retrieved from http://projects.spring.io/spring-integration/: http://projects.spring.io/spring-integration/

Stephenson, P. (2014, November 15). Emerging products: Data classification. *SC Magazine: For IT Security Professionals (15476693), 25*(11), p. 42. Retrieved Febraruary 20, 2016

The Washington Post. (2013). NSA Secrets. In G. Miller, *Edward Snowden, Bradley Manning and the risk of the low-level, tech-savvy leaker.* New York: Diversion Books.

Wah, M. (2016, April 29). *819767_EMC_a412402_RSA_DLP_95.pdf.* Retrieved from arrowecs.hu: http://www.arrowecs.hu/products/rsa/doc/819767_EMC_a412402_RSA_DLP_95.pdf

Williams, B. R. (2015). *PCI DSS 3.1.* Waltham, MA: Syngress.

Vormetric. (2015, September 1). *CW_GlobalReport_2015_Insider_threat_Vormetric_Single_Pages_010915.* Retrieved February 18, 2016, from enterprise-encryption.vormetric.com: http://enterprise-encryption.vormetric.com/rs/vormetric/images/CW_GlobalReport_2015_Insider_threat_Vormetric_Single_Pages_010915.pdf