CloudDLP: Transparent and Scalable Data Sanitization for Browser-Based Cloud Storage

PEIYI HAN1,2, CHUANYI LIU2,3, JIAHAO CAO4, SHAOMING DUAN2, HEZHONG PAN1, ZEKUN CAO5,6, AND BINXING FANG2,3
1School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China
2School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen 518055, China
3Peng Cheng Laboratory, Shenzhen 518055, China
4Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
5CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
6School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China
Corresponding author: Chuanyi Liu (liuchuanyi@hit.edu.cn)

Received March 5, 2020, accepted April 2, 2020, date of publication April 6, 2020, date of current version April 22, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2985870

ABSTRACT

Browser-based cloud storage services are still broadly used in enterprises for online sharing and collaboration. However, sensitive information in images or documents may be easily leaked outside trusted enterprise on-premises due to such cloud services. Existing solutions to prevent data leakage in cloud storage services either limit many functionalities of cloud applications or are difficult to be scaled to various cloud applications. In this paper, we propose CloudDLP, a transparent and scalable approach for enterprises to automatically sanitize sensitive data in images and documents with various browser-based cloud applications. CloudDLP is deployed as an internet gateway within the premises of an enterprise using JavaScript injecting techniques and deep learning methods to sanitize sensitive premise data. It neither compromises the user experience nor significantly affects application functionalities in browser-based cloud storage services. We have evaluated CloudDLP with a number of real-world cloud applications. Our experimental results show that it can achieve automatic data sanitization with cloud storage services while preserving most functionalities of cloud applications.

INDEX TERMS Data loss prevention, data sanitization, data security, cloud storage.

I. INTRODUCTION

Browser-based cloud storage services are still widely used in enterprises, such as Dropbox [1], Box [2], and Salesforce [3]. They provide great conveniences for online sharing and collaboration, save the cost for data storage, and maintain high data reliability, which brings much benefits for enterprises. However, sensitive information stored in these services may be potentially leaked or shared outside of trusted premises due to vulnerabilities or misuses in insecure clouds. For example, Google discovered a flaw in Google Drive that may grant unauthorized parties access to a subset of documents under certain circumstances [4]. Dropbox and Box were also affected by similar security issues, which can allow third parties to discover private file transfer links [5]. Moreover, researchers have identified several defects and misuses in Amazon Simple Storage Service (S3) that can leak military secrets [6] and private medical data [7]. Therefore, it is critical to protect sensitive information from leaking out of enterprise on-premises to insecure clouds.

Recently, various approaches have been provided to prevent data leakage in the browser-based cloud storage. Many studies [8]–[12] apply the Data Loss Prevention (DLP) technology to detect and filter out sensitive information using massive regular expressions, custom keywords, or domain specific entity corpora. Message-Guard [13] creates a file upload overlay using HTML iFrames to effectively protect data at the cost of significantly reducing functionalities of cloud applications, which is not transparent to diversified functionalities of cloud applications. Moreover, the Cloud Access Security Broker (CASB) solutions [10],

VOLUME 8, 2020 This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/ 68449
[14], [15] can provide protection of sensitive information. However, they are difficult to be scaled to various applications due to the need of reversing complex application-specific protocols.

In this paper, we present CloudDLP, a transparent and scalable solution to protect sensitive data for browser-based cloud applications. CloudDLP allows enterprises and users to upload encrypted or sanitized sensitive data while maintaining most of the original functionalities of the cloud applications. It is deployed as an internet gateway within the premises of an enterprise and sets between cloud applications and users. When users upload their sensitive data files to the browser-based cloud application, CloudDLP can transparently capture the data files in the requests. Sensitive information in the data files is then detected and sanitized.

CloudDLP is designed for preventing data leakage from malicious or compromised cloud applications. A key challenge in developing CloudDLP is to provide the scalability for various cloud applications. If we adapt CloudDLP to cloud applications one by one, it will be time-consuming and error-prone. To solve the problem, CloudDLP leverages the JavaScript injecting technique that instruments JavaScript snippets to the web application pages. Consequently, file requests containing sensitive data to various cloud applications can be detected.

To detect a wide range of sensitive information in the images and documents that may be shared unintentionally, CloudDLP extracts textual data from images and documents with deep learning methods. CloudDLP classifies images and documents into different sensitivity levels and assigns proper protection policies. To preserve the application functionality as much as possible, CloudDLP develops an automatic sanitization method based on deep learning. Compared to traditional DLP approaches, our automatic detection and sanitization method is transparent to cloud applications and shows superior performance. With CloudDLP, enterprises can effectively prevent data leakage without compromising many functionalities of cloud applications.

To validate the feasibility of CloudDLP, we have implemented and evaluated it in ten real-world browser-based cloud storage services, such as email, storage, and office software. The experiments on these applications successfully demonstrate the effectiveness and scalability of the proposed approach. Moreover, the results show that the accuracy of the image sanitization can achieve 93.4%, which is acceptable for sensitive information extraction. The accuracy of the document sanitization can achieve 97.9%. Therefore, CloudDLP is able to protect sensitive data with various browser-based cloud applications. This paper, which is extended from our previous paper [16], has the following contributions:

- We propose CloudDLP, a system that provides transparent and scalable detection and sanitization for sensitive data with various browser-based cloud storage services.
- We propose deep learning methods together with the JavaScript injecting technique to detect and sanitize sensitive data in images and textual documents, which preserves sensitive data while maintaining functionalities of cloud applications as much as possible.

We have implemented and tested CloudDLP with various popular cloud applications. Our experimental results demonstrate the effectiveness of CloudDLP with various cloud applications.

The remainder of the paper is organized as follows: we review the related work in Section II. We present CloudDLP in Section III. We present the evaluation results and case studies in Section IV. Finally, we conclude the paper in Section V.

II. RELATED WORK

Many solutions to protect sensitive data for cloud storage services have been proposed. There are three categories of related studies: Data Loss Prevention, Cloud Access Security Broker, and other work.

A. DATA LOSS PREVENTION

Data Loss Prevention is the technology of detecting and preventing data breaches. Several methods such as SPIRION [8] and CUSpider [9] adopt pre-defined keywords and regular expressions to detect sensitive information. However, these methods not only need lots of rules but also have a lower detection precision due to many false positives. Costante et al. [17] presented a data loss prevention framework based on signature-based and anomaly-based methods. The framework uses a machine learning algorithm to check user behaviors and builds malicious behavior signatures library. Gomez-Hidalg et al. [18] proposed achieving data loss prevention by Named Entity Recognition method. Unfortunately, images cannot be protected by this approach. Ong et al. [19] introduced a system that is able to detect sensitive information in the document via a deep learning model. The detection model predicts sensitive information base on semantic context analysis. These systems are unable to detect sensitive information in images or other non-document data. They are difficult to apply in practice because their performance is fairly disappointing.

B. CLOUD ACCESS SECURITY BROKER

CASB [14] is advocated by Gartner as a leading technology in cloud security. It is an approach based on a proxy server launched by cybersecurity companies such as CipherCloud [10] and SkyHigh Network [15], to protect crucial data. CASB is deployed as a proxy sets between cloud applications and users. The proxy intercepts, detects and encrypts sensitive data through protocol analysis. However, they have to adopt various cloud applications by reverse engineering network protocols, which is time-consuming and labor-intensive. Therefore, this approach is difficult to be scaled to various applications.

C. OTHER WORK

Song has developed ShadowCrypt [11] which runs as a browser extension, to provide textual data protection for existing cloud applications. However, ShadowCrypt can not
support data files such as documents, images and so on. Mimesis Aegis [12] is applied in mobile applications and provides sensitive data isolation through a conceptual layer. It also cannot provide data file protection. CryptDB [20] is deployed as a proxy between the database server and the application server, for encrypting user confidential data. It can effectively prevent curious DBA from learning private data. But it is only applied in the database. Mylar [21] is used to protect application data based on the Meteor JavaScript framework for web developers. However, Mylars lacks of compatibility and cannot support data analysis. Virtru [22] is an email encryption system for avoiding webmail leakage. It is suitable for email service and is difficult to scale to the new applications.

III. SYSTEM ARCHITECTURE

A. DESIGN GOALS

In this paper, we aim to develop methods to protect sensitive information of images and documents from leaking out of enterprise on-premises to insecure cloud storage via browser-based interfaces. Developing a approach for browser-based cloud storage services is usually not an easy task, due to the following requirements:

1. **Security.** The solution needs to ensure that sensitive data is protected before leaving enterprise on-premises to the cloud.
2. **Usability.** The method needs better preserve user experiences and application functionalities, such as document previewing and editing.
3. **Scalability.** The solution must be easy to maintain and be highly scalable for various applications.

B. THREAT MODEL

CloudDLP should be deployed at the edge of an enterprise network, where main security restricted policies can keep most attackers out. We assume that an internal enterprise network is secure and trustworthy. But cloud storage service providers may be malicious. Sometimes, they may compromise the security of valuable user data due to commercial interests, legal requirements, or when they are compromised. Meanwhile, the client-side application code and the middleware on the network channel may also be exploited to exfiltrate sensitive information.

Additionally, we assume the operating systems, browsers, and network devices inside an enterprise firewall are trusted, which is a practical trusted zone in most organizations. CloudDLP does not provide protection against side-channel attacks.

C. CloudDLP ARCHITECTURE

In this section, we present the architecture of CloudDLP and briefly describe the functionality of each component. The architecture of CloudDLP is shown in Fig. 1. CloudDLP consists of five components components: (a) Interceptor, (b) Parser, (c) Classifier, (d) Sanitization, and (e) Packer.

1) **Interceptor.** It is able to intercept HTTP/HTTPS traffic from users on the premises to the cloud. The Interceptor can also as a “man-in-the-middle” to intercept TLS connections, which is authorized by the enterprise. By intercepting the traffic between users and the cloud, the interceptor can effectively prevent sensitive information from leakage to outside enterprises. Moreover, it is also responsible for injecting the JavaScript snippets into the web pages of cloud applications. The snippets used to override JavaScript native API, can intercept all the XMLHttpRequest and identify file uploading requests.

2) **Parser.** The Parser is used to parse application protocols, analyze semantic content of protocol and extract file content. It obtains data buffered for a user session and analyzes the request content format (including key-value, multi-part, etc.) to extract document fields.
3) **Classifier.** The Classifier is responsible for sensitive data categorization, which helps categorize stored data by sensitivity and business impact and takes different level protection policies. The Classifier adopts deep learning models to classify confidential data.

4) **Sanitizer.** It is responsible for detecting and sanitizing sensitive information of images and documents. This module minimizes the risk of data leakage while still keeping documents and images available for preserving functionalities such as document editing, document preview, and thumbnail preview. We also develop methods that identify sensitive data of images and textual documents via a deep learning model. Thus we adopt the scene text reading approach based on convolutional neural network (CNN) and recurrent neural network (RNN) to detect sensitive data. CloudDLP associates the discovery of sensitive data in text to the recognition of Named Entities in Natural language processing (NLP). Therefore, Sanitizer ensures that detection and redaction of private information while preserving origin functionality.

5) **Packer.** The Packer can assemble the redacted images and documents into HTTP/HTTPS requests. Then CloudDLP would send the requests which contain protected data, to the cloud server.

CloudDLP workflow is shown in Algorithm 1. when an enterprise user accesses a cloud service by CloudDLP, CloudDLP receives the requests. The interceptor component can inject the JavaScript snippets to the web pages in the requests. The Snippets would be instrumented at the head of the web pages. The native JavaScript API such as XMLHttpRequest and FileReader is overridden by the snippets. Then, the snippets start to monitor the connections and file operations invoked by the JavaScript. The file requests are identified by the associated API functions and object methods. Next, the parser parses the request format and extracts the file contents. The data file is classified as sensitive by the classifier and is sanitized by automatic image sanitization or automatic document sanitization. In the end, the packer reassembles the sanitized files into the associated requests then forwards them to the cloud server.

### Algorithm 1 The Main CloudDLP Algorithm

| Input: B: the browser | CSS: the Cloud Storage Service | C: CloudDLP |
|-----------------------|--------------------------------|-------------|
| while a connection c from B do |
| if c.request contains html page then |
| c.response ← C.inject_js_snippet(c.response) |
| if c.request is identified as file uploading then |
| file ← C.parse(c.request) |
| sensitive ← C.classify(file) |
| file ← C.sanitize(file) |
| c.request ← C.pack(c.request, file) |
| send(c) |

D. AUTOMATIC IDENTIFICATION OF FILES

In order to fulfill the specified design goals, CloudDLP supports automatically new and existing applications through recognizing file requests from the cloud. In this section, we first discuss two unsatisfactory methods and present automatic identification of files via the dynamic analysis of JavaScript.

**Strawman 1:** The traditional method leverages regular expression matching to match file upload requests. Facing a large number of cloud applications, CloudDLP needs to analyze application protocols and maintain a large number of regular matching rules. In addition, once the application protocol changes, the corresponding rules need to be changed in time. Therefore this method is laborious and less stable in practice.

**Strawman 2:** We could adopt inter-procedural string analysis [23] to extract the corresponding file upload API from the JavaScript code of cloud applications. In fact, this method is difficult to extract accurate file APIs since JavaScript code usually is compressed and obfuscated by the cloud providers. When the inter-procedural strong analysis cannot extract file operation APIs, the proposed system will fail.

In reality, file transmission is always accompanied by invoking an XMLHttpRequest object in cloud applications. All contemporary browsers have a built-in XMLHttpRequest object, which defines a programming interface for data file transfer. XMLHttpRequest provides several ways to access data file through a file objects such as Blob [24], File, FormData [25], or ArrayBuffer [26]. A file would be stored in the above file objects of JavaScript. These objects stand for a data file object. Blob and File mean the immutable file data. FormData can be consist of Blob/File object and ArrayBuffer represents the binary raw data read from a file. CloudDLP hooks and overrides the XMLHttpRequest JavaScript API via injecting JavaScript snippets. The argument type of API can be checked by the snippets for identifying file requests. Then it is easy to monitor, intercept and modify requests from XMLHttpRequest.

E. AUTOMATIC IMAGE SANITIZATION

To meet our security and usability goal, CloudDLP needs to detect sensitive information hidden in the images. To achieve this challenge, CloudDLP adopts a scene-text reading solution for detecting automatically sensitive information. The solution is divided into two phases: text detection and text recognition. The text detection phase is used to locate the area of sensitive text from the images. The text recognition phase is used to transform sensitive text areas detected in the previous phase into editable text which can be used for text processing. Moreover, the model structure is presented in Fig. 2:

**Text Detection Process:** Most research studies of the text detection model are designed to detect extremely long...
texts [27]–[29] and the multi-oriented text [30]–[32]. Since the Connectionist Text Proposal Network (CTPN) model [33]–[35] performs well on sensitive texts in images which are always horizontal and low-pixel, CloudDLP applies this model to text detection. The process of text detection is as follows:

1. An input image which contains sensitive information, is divided into small blocks with 16 × 16 pixels.
2. VGG16 consisting of five convolutional neural networks (CNN), extracts features from the image. The size of the features is $W \times H \times C$.
3. An additional convolution layer is used to convert the features into sliding windows. Then each sequential windows connected to a Bi-directional LSTM (BiLSTM) for analyzing these feature sequences. BiLSTM is designed for processing sequence data, and it combines all the features.
4. The BiLSTM layer is connected to a 512 dimension fully-connected layer. The output layer with 60 hidden units predicts the center point coordinate, the fixed-width of the proposal and scores which whether the proposal is text or non-text.

Text Recognition Process: The output of some text recognition models is often corrected by a word dictionary called ‘lexicon’. Although this method can improve the recognition accuracy to some extent, there is no preset sensitive data dictionary in the data protection scenario on the cloud. At the same time, sensitive information is context-sensitive. Therefore, Phase combines CNNs and RNNs into a single network, referred to as a convolutional Recurrent Neural Network (CRNN) [36]. The model is dedicated to recognizing horizontal texts and perform well without lexicon. Although this method can improve the accuracy to a certain extent, there is no preset sensitive data dictionary in the data protection scenario on the cloud. The CRNN consists of four parts:

1. All of the text regions are scaled to a height of 32 pixels by bilinear sampling.
2. At the top of the architecture, the above text regions are fed to convolutional layers for extracting features and transforming into sequential vector representations. The max-pooling windows of the CNN are 1 × 2, which in order to directly require feature sequences from the text regions.
3. The features of the last convolutional layer are fed to the bidirectional LSTM layer.
4. The final recurrent layer outputs are read by a single feedforward layer, which estimates a label for each vector then outputs a label sequence. The label sequence is computed by Connectionist Temporal Classification (CTC) [37] and to return the final result.

Next, the final result which is the recognized text, can be handled by the Named Entity Recognition model mentioned in Section 3.7, when it belongs to a sensitive entity.

F. TEXTUAL DOCUMENT CLASSIFICATION

Textual document classification provides a convenient way for CloudDLP to determine and assign proper protection policy to the data it processes. Content-sensitive and important documents need to be encrypted or sanitized, while content-insensitive and less-sensitive documents can be released without any processing. This method can greatly reduce the number of documents it processes and improves its performance. To tackle this problem, we adopt TextCNN [38] to achieve text topic classification, and then to grade the sensitivity of the document based on the text topic. The TextCNN model architecture, shown in Fig. 3. A document consists of multiple sentences and the i-th word in the sentence is represented by the k-dimensional word vector $x_i \in \mathbb{R}^k$. It is worth briefly mentioning that the word vector is learned from [39]. A sentence of length $n$ is represented as $x_{1:n} = x_1 \oplus x_2 \oplus \ldots \oplus x_n$, where $\oplus$ is the concatenation operator. A window of $h$ words $x_{i:i+h-1}$ from the sentence would be applied to a convolution operation involves a filter for producing a new feature $c_i$. Then the filter is applied to each window of words in each sentence for producing feature maps.
A max-over-time pooling operation over the feature maps and take the maximum value. These features are passed to a last fully connected softmax layer. The probability distribution over labels is output from the fully connected layer.

### G. AUTOMATIC TEXTUAL DOCUMENT SANITIZATION

One promising way of protecting files uploaded to the cloud is encryption. However, file encryption can cause cloud application functionality such as online document editing and preview to fail. Document sanitization can be capable of redacting sensitive data and preserve the integrity of the document structure. Hence, sanitizing automatically sensitive information is a big challenge.

CloudDLP adopts BERT (Bidirectional Encoder Representations from Transformers) [39] published by researchers at Google AI Language Group, to identify into sensitive categories such as person names, organization, places and so on. BERT is pre-trained on a large corpus of unlabelled text which includes the entire Wikipedia (that’s about 2,500 million words) and a book corpus (800 million words). And we apply BERT to automatic textual document sanitization via transfer learning on our datasets. Thus we employ a pre-trained BERT with Conditional Random Fields (CRF) architecture to sensitive entity recognition. The network architecture is illustrated in Fig. 4. The model architecture is composed of the BERT and the CRF. A document needs to break down into sentences and split into tokens. The input of BERT allows input sequences constructed by summing the corresponding token, segment and position embeddings. BERT is a multi-layer bidirectional transformer encoder, which consists of 12 transformer blocks and has 12 self-attention heads. The transformer process words in relation to all the other words in a sentence, rather than one-by-one in order. It can learn richer semantic information and perform well on the sensitive entity recognition task. BERT is followed by a linear-chain conditional random field (CRF) [40], which is used to search the labeling sequence with the highest probability. And the most likely sequence would be obtained by Viterbi decoding of the CRF.
FIGURE 5. Text detection and recognition network.

Wild) [44] and ICPR Text Detection [45]. The text recognition model is trained with synthetic datasets combined with actual data protection scenarios. The synthetic datasets are generated from suitable fonts, backgrounds, and corpus, which make the datasets more realistic. The convolutional neural networks are trained by Stochastic Gradient Descent (SGD) and recurrent neural networks are trained by Back-Propagation Through Time (BPTT) in these models. ADADELTA [46] is applied to control the learning rate and batch normalization is used to speed up the training process. The details of network parameters is presented in Fig. 5, where \( s \) is step size, \( p \) is padding size, \( c \) is number of channels, and \( k \) is kernel size of the convolution layer.

The experiments are conducted on evaluating the performance of text detection model via DetEval algorithm [47] and evaluating the accuracy of text recognition model by computing Edit Distance. The details of network parameters is presented in Fig. 5, where \( s \) is step size, \( p \) is padding size, \( c \) is number of channels, and \( k \) is kernel size of the convolution layer.

The experiments are conducted on evaluating the performance of text detection model via DetEval algorithm [47] and evaluating the accuracy of text recognition model by computing Edit Distance. These experiments are evaluated on three mutually-exclusive data sets, image datasets collected from three different widely used cloud applications. The results of our experiments are presented in Table. 1. The accuracy of the automatic image sanitization model is about 93.4% on data protection scenario. The average detection latency per time step is 4.686 s and the average recognizable latency per time step is 0.415 s. The above cost time is acceptable in practice. The actual effect of the automatic image sanitization component is shown in Fig. 6. It can detect and sanitize sensitive information such as person name, age, and ID number in medical images.

B. TEXTUAL DOCUMENT CLASSIFICATION PERFORMANCE

We scraped the publication document data from the websites and labeled the document. Then we got seven categories: “Sports”, “Technology”, “Stocks”, “Finance”, “Government Affairs”, “Law”, and “Entertainment”. Each category has 80,000 training data, 10,000 verification data, and 10,000 tests data. Next, we delete the special characters in the datasets, use the stuttering word segmentation tool to segment the words and remove the stop words. In addition, we initialize word vectors with publicly available word vectors that were trained on 40 epochs over a 3.3 billion word corpus from Google, which is a popular method to improve performance. The experiment computed precision, recall and F1 rates as evaluation metrics. We use \( N_1 \) to denote the number of correctly identified as \( c \) category, \( N_2 \) to denote the number of identified as \( c \) category, \( N_3 \) to denote the total number of all categories, \( N_4 \) to denote the total number of correctly identified for each category. The metrics can be calculated as follows:

\[
\text{Precision} = \frac{N_1}{N_2} \quad (1)
\]

\[
\text{Recall} = \frac{N_1}{N_3} \quad (2)
\]

\[
F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)
\]

\[
\text{Micro\_Precision} = \frac{N_4}{N_3} \quad (4)
\]

\[
\text{Micro\_Recall} = \frac{N_4}{N_3} \quad (5)
\]

\[
\text{Micro\_F}_1 = \frac{2 \times \text{Micro\_Precision} \times \text{Micro\_Recall}}{\text{Micro\_Precision} + \text{Micro\_Recall}} \quad (6)
\]

Results of our model are listed in Table. 2. Our model can achieve 0.9762 Micro\_Precision on textual document classification.

C. TEXTUAL DOCUMENT SANITIZATION PERFORMANCE

The 2014 i2b2 de-identification challenge data set [48] can be used to train and evaluate sensitive entity recognition model. CloudDLP needs to detect and redact sensitive health information (PHI) in the medical records. The i2b2-PHI categories are presented in Table. 3. The dataset has 1,304 patient progress notes for 296 diabetic patients. And it consists of 56,348 sentences with 984,723 separate tokens. 41,355 of the tokens are separate PHI tokens, which represent 28,867 separate PHI instances. The standard metrics used to measure the performance of the model, such as precision, recall, and F1 as defined in Equation 7-9. We use \( N_1 \) to denote the number of correctly identified PHI tokens, \( N_2 \) to denote the number of identified as PHI tokens, \( N_3 \) to denote the total

### TABLE 2. Textual document classification performance.

| Category   | Precision | Recall | F1    |
|------------|-----------|--------|-------|
| Sports     | 0.9970    | 0.9944 | 0.9957|
| Technology | 0.9814    | 0.9808 | 0.9811|
| Stock      | 0.9535    | 0.9482 | 0.9509|
| Financial  | 0.9392    | 0.9370 | 0.9381|
| Government | 0.9793    | 0.9860 | 0.9827|
| Law        | 0.9998    | 1.0000 | 0.9999|
| Entertainment | 0.9877   | 0.9946 | 0.9912|
| Average    | 0.9762    | 0.9762 | 0.9762|

### TABLE 3. i2b2-PHI categories.

| Category | Description |
|----------|-------------|
| Sports   | Activity    |
| Technology | Diagnosis  |
| Stock    | Treatment   |
| Financial | Medication  |
| Government | Observations  |
| Law      | Procedures   |
| Entertainment | Laboratory  |
The BERT-CRF model has 12 layers of Transformer Blocks, 768 hidden size, 12 self-attention heads and 110 million parameters. The experiment is evaluated on i2b2-PHI categories for the i2b2 data set based on token-level labels. Table 4 summarizes the performance of our model on i2b2-PHI categories. As we can see, the model achieves an impressive precision 0.9796 and the recall 0.9836. Moreover, Fig. 7 shows the actual effect of the automatic textual document sanitization model, which detects and sanitizes sensitive information.

**D. CASE STUDY**

We tested CloudDLP on a wide variety of typical cloud applications that store data files, which is the focus of CloudDLP. The contribution of CloudDLP is providing transparent and scalable detection and sanitization and putting the enterprise back in control of their data. In this experiment, we test CloudDLP and observe the potentially affected functionalities including search, online document editing, document previews, and thumbnail previews. We are able to use CloudDLP to protect sensitive critical data while retaining the prominent functionality of these applications. We tested CloudDLP with typical browser-based cloud storage applications such as Gmail, Dropbox, Box, OneDrive, Google Drive, and Mega.nz. Sensitive information of files uploaded to the cloud, can be successfully detected and sanitized. sharing of documents, document previewing and editing can still be used normally since the sensitive information is sanitized.

**TABLE 3. PHI categories on i2b2.**

| Category       | i2b2          |
|----------------|---------------|
| Name           | Patient, Doctor, Username |
| Profession     | Profession |
| Location       | Street, City, State, Country, Zip, Hospital, Organization |
| Age            | Age           |
| Date           | Date          |
| Contact        | Phone, Fax, Email, URL, IP Address |
| ID             | Medical Record, ID No, SSN, License No |
without destroying the usability of documents or images. And the cloud application could parse and handle these protected files. On using CloudDLP with office applications, namely Salesforce, Google Docs, and Slack, The uploaded files of Salesforce Salesforce and Google Docs. Google Docs can be sanitized and protected by CloudDLP. Since the data which is numeric type is sanitized, importing the protected files into Salesforce may affect statistical analysis in the trading reports.

V. DISCUSSION

We note several limits of validity of the technique. CloudDLP is works for browser-based cloud storage applications, and is ineffective if client-based cloud applications are used. In addition, the sanitization may be reversed under certain circumstances. We will consider more secure sanitization methods.

VI. CONCLUSION

In this paper, we present CloudDLP, a transparent and scalable approach to protect sensitive data for the browser-based cloud applications. It can help enterprises effectively prevent critical data leakage. In addition, our experimental results show that it can achieve automatic data sanitization with cloud storage services while preserving most functionalities of cloud applications. Moreover, CloudDLP can support practical use in many real-world applications.

REFERENCES

[1] Dropbox: Cloud Storage Service. Accessed: Jan. 21, 2020. [Online]. Available: https://www.dropbox.com/

[2] Box: Cloud Storage Service. Accessed: Jan. 21, 2020. [Online]. Available: https://www.box.com/

[3] Salesforce: Customer-Relationship Management Service. Accessed: Jan. 21, 2020. [Online]. Available: https://www.salesforce.com/

[4] Google Drive Update to Protect to Shared Links. Accessed: Jan. 21, 2020. [Online]. Available: https://security.googleblog.com/2014/06/google-drive-update-to-protect-to.html

[5] Your Sensitive Information Could be at Risk: File Sync and Share Security Issue. Accessed: Jan. 21, 2020. [Online]. Available: https://blogs.intralinks.com/2014/05/sensitive-information-risk-file-sync-share-security-issue/

[6] Top Defense Contractor Left Sensitive Pentagon Files on Amazon Server With no Password. Accessed: Jan. 21, 2020. [Online]. Available: http://gizmodo.com/top-defense-contractor-left-sensitive-pentagon-files-on-1795669632.

[7] Patient Home Monitoring Service Leaks Private Medical Data. Accessed: Jan. 21, 2020. [Online]. Available: https://mackeepersecurity.com/post/patient-home-monitoring-service-leaks-private-medical-data-online

[8] Spirion. Accessed: Jan. 21, 2020. [Online]. Available: https://www.spirion.com/

[9] Cuspider. Accessed: Jan. 21, 2020. [Online]. Available: https://www.cypercloud.com/

[10] Ciphercloud: Cloud Services Adoption While Ensuring Security, Compliance and Control. Accessed: Jan. 21, 2020. [Online]. Available: https://www.ciphercloud.com/

[11] W. He, D. Akhawe, S. Jain, E. Shi, and D. Song, "ShadowCrypt: Encrypted Web applications for everyone,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., New York, NY, USA, 2014, pp. 1028–1039, doi: 10.1145/2660267.2660326,

[12] B. Lau, S. Chung, C. Song, Y. Jang, W. Lee, and A. Boldyrev, “Mimesis aegis: A mimicry privacy shield–a system’s approach to data privacy on public cloud,” in Proc. 23rd Secur. Symp., 2014, pp. 33–48.

[13] S. Ruoti, J. Andersen, T. Monson, D. Zappala, and K. Seamons, “MessageGuard: A browser-based platform for usable, content-based encryption research,” 2015, arXiv:1510.08943. [Online]. Available: http://arxiv.org/abs/1510.08943

[14] Cloud Access Security Brokers. Accessed: Jan. 21, 2020. [Online]. Available: https://www.gartner.com/it-glossary/cloud-access-security-brokers/

[15] Skyhigh: 9 9 Cloud Computing Security Risk Every Company Faces. Accessed: Jan. 21, 2020. [Online]. Available: https://www.skyhighnetworks.com/cloud-security-blog/9-cloud-computing-security-risks-every-company-faces/

[16] C. Liu, P. Han, Y. Dong, H. Pan, S. Duan, and B. Fang, “CloudDLP: Transparent and automatic data sanitization for browser-based cloud storage,” in Proc. 28th Int. Conf. Comput. Commun. Netw. (ICCCN), Jul. 2019, pp. 1–8.

[17] E. Costante, D. Fauri, S. Etalle, J. D. Hartog, and N. Zannone, “A hybrid framework for data loss prevention and detection,” in Proc. IEEE Secur. Privacy Workshops (SPW), May 2016, pp. 324–333.

[18] J. M. Gomez-Hidalgo, J. M. Martin-Arteau, J. Nieves, I. Santos, F. Brezo, and P. G. Bringas, “Data leak prevention through named entity recognition,” in Proc. IEEE 2nd Int. Conf. Social Comput., Aug. 2010, pp. 1129–1134.

[19] Y. J. Ong, M. Qiao, R. Routray, and R. Raphael, “Context-aware data loss prevention for cloud storage services,” in Proc. IEEE 10th Int. Conf. Cloud Comput. (CLOUD), Jun. 2017, pp. 399–406.

[20] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, “CryptDB: Protecting confidentiality with encrypted Query processing,” in Proc. 29rd ACM Symp. Oper. Syst. Princ. (SOSP), 2011, pp. 85–100.
[21] P. Grubbs, R. McPherson, M. Naveed, T. Ristenpart, and V. Shmatikov, “Breaking Web applications built on top of encrypted data,” in Proc. ACM SIGSAC Conf. Comput. Commun. Secur., 2016, pp. 157–172.

[22] Virtru: Email Encryption and Data Security for Business Privacy. Accessed: Jan. 21, 2020. [Online]. Available: https://www.virtru.com

[23] E. Wittern, A. T. T. Ying, Y. Zheng, J. Dolby, and J. A. Laredo, “Safely checking Web API requests in JavaScript,” in Proc. IEEE/ACM 39th Int. Conf. Softw. Eng. (ICSE), May 2017, pp. 244–254.

[24] JavaScript API-Blob. Accessed: Feb. 3, 2020. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Blob

[25] JavaScript API-ArrayBufferView. Accessed: Feb. 3, 2020. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/ArrayBufferView

[26] JavaScript API-FormData. Accessed: Feb. 3, 2020. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/FormData

[27] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo, “R2CNN: Rotational region CNN for orientation robust scene text detection,” 2017. arXiv:1706.09579. [Online]. Available: http://arxiv.org/abs/1706.09579

[28] B. Shi, X. Bai, and S. Belongie, “Detecting oriented text in natural images by linking segments,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 2550–2558.

[29] D. Deng, H. Liu, X. Li, and D. Cai, “Pixellink: Detecting scene text via instance segmentation,” in Proc. 32nd AAAI Conf. Artif. Intell., Apr. 2018, pp. 6773–6780.

[30] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, “Multi-oriented text detection with fully convolutional networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 4159–4167.

[31] C. Yao, X. Bai, and W. Liu, “A unified framework for multi-oriented text detection and recognition,” IEEE Trans. Image Process., vol. 23, no. 11, pp. 4737–4749, Nov. 2014.

[32] Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao, “Oriented response networks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 519–528.

[33] Z. Tian et al., “Detecting text in natural image with connectionist text proposal network,” in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2016.

[34] D. Wu, R. Wang, P. Dai, Y. Zhang, and X. Cao, “Deep strip-based network with cascade learning for scene text localization,” in Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR), Nov. 2017, pp. 826–831.

[35] X. Zhu, Y. Jiang, S. Yang, X. Wang, W. Li, P. Fu, H. Wang, and Z. Luo, “Deep residual text detection network for scene text,” in Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR), Nov. 2017, pp. 807–812.

[36] B. Shi, X. Bai, and C. Yao, “An End-to-End trainable neural network for image-based sequence recognition and its application to scene text recognition,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 11, pp. 2298–2304, Nov. 2017.

[37] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in Proc. 23rd Int. Conf. Mach. Learn. (ICML), 2006, pp. 369–376.

[38] Y. Kim, “Convolutional neural networks for sentence classification,” 2014. arXiv:1408.5882. [Online]. Available: http://arxiv.org/abs/1408.5882

[39] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018. arXiv:1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805

[40] C. Sutton, K. Rohanimanesh, and A. McCallum, “Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data,” in Proc. Twenty-first Int. Conf. Mach. Learn. (ICML), 2004.

[41] Squid. Accessed: Feb. 10, 2020. [Online]. Available: http://www.squid-cache.org/

[42] Tensorflow. Accessed: Feb. 10, 2020. [Online]. Available: https://www.tensorflow.org/

[43] Pytorch. Accessed: Feb. 10, 2020. [Online]. Available: https://pytorch.org/

[44] B. Shi, C. Yao, M. Liao, M. Yang, P. Xu, L. Cui, S. Belongie, S. Lu, and X. Bai, “ICDAR2017 competition on reading chinese text in the wild (RCTW-17),” in Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR), Nov. 2017, pp. 1429–1434.

[45] Icpr Text Detection Dataset. Accessed: Feb. 10, 2020. [Online]. Available: https://tianchi.aliyun.com/competition/entrance/231651/information

[46] M. D. Zeiler, “ADADELTA: An adaptive learning rate method,” 2012, arXiv:1212.5701. [Online]. Available: http://arxiv.org/abs/1212.5701

[47] C. Wolf and J.-M. Jolion, “Object count/area graphs for the evaluation of object detection and segmentation algorithms,” Int. J. Document Anal. Recognit. (JIDAR), vol. 8, no. 4, pp. 280–296, Sep. 2006.

[48] I2B2 Dataset. Accessed: Feb. 10, 2020. [Online]. Available: https://www.i2b2.org/NLP/DataSets/

PEIYI HAN was born in Xiaoyi, Shanxi, China, in 1992. He received the M.S. degree in computer science and technology from the Beijing University of Posts and Telecommunications, Beijing, in 2016, where he is currently pursuing the Ph.D. degree in information security. His research interests include cloud computing, searchable encryption, and data security.

CHUANYI LIU was born in Leshan, Sichuan, China, in 1982. He received the M.S. and Ph.D. degrees in computer science and technology from Tsinghua University, Beijing, in 2010. From 2007 to 2008, he was a Visiting Scholar in computer and information science with the University of Minnesota. He currently works at the Harbin Institute of Technology, Shenzhen, China. His research interests include cloud computing, cloud security, mass storage systems, data protection, and data security.

JIAHAO CAO was born in Hengshui, Hebei, China, in 1993. He received the B.Eng. degree from the Beijing University of Posts and Telecommunications, in 2015. He is currently pursuing the Ph.D. degree with Tsinghua University. His current research interests include cloud computing, cloud security, SDN, and NFV security.

SHAOMING DUAN was born in Shaoyang, Hunan, China, in 1994. He received the M.S. degree in computer technology from the Harbin Institute of Technology, in 2018. He is currently pursuing the Ph.D. degree with the Cyberspace Security Center, College of Computer Science and Technology, Harbin Institute of Technology (Shenzhen). His current research interests include data protection and machine learning.

HEZHONG PAN was born in Benxi, Liaoning, China, in 1991. He received the M.S. degree in computer science and technology from the Beijing University of Posts and Telecommunications, Beijing, in 2015, where he is currently pursuing the Ph.D. degree in information security. His research interests include cloud computing, searchable encryption, and data security.
ZEKUN CAO was born in Anqing, Anhui, China, in 1995. He received the B.E. degree in computer science and technology from Anhui University, Anhui, in 2017. He is currently pursuing the M.S. degree in information security with the University of Chinese Academy of Sciences, Beijing. His research interests include machine learning and data security.

BINXING FANG was born in Wanning, Jiangxi, China, in 1960. He received the M.S. degree in computer science and technology from Tsinghua University, Beijing, in 1984, and the Ph.D. degree in computer science and technology from the Harbin Institute of Technology, Harbin, in 1989. He is currently a member of the Chinese Academy of Engineering. His current research interests include computer networks, information and network security, and content security.