applied sciences

Article

SA-UBA: Automatically Privileged User Behavior Auditing for Cloud Platforms with Securely Accounts Management

Hezhong Pan [1], Peiyi Han [2,*], Xiayu Xiang [3], Shaoming Duan [2] and Chuanyi Liu [2,3,*]

[1] School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2] School of Computer Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen 518055, China
[3] Peng Cheng Laboratory, Shenzhen 518055, China
[*] Correspondence: hanpeiyi@hit.edu.cn (P.H.); liuchuanyi@hit.edu.cn (C.L.); Tel.: +86-185-1063-1118 (P.H.); +86-150-1009-3819 (C.L.)

Abstract: Cloud platforms allow administrators or management applications with privileged accounts to remotely perform privileged operations for specific tasks, such as deleting virtual hosts. When privileged accounts are leaked and conduct dangerous privileged operations, severe security problems will appear on cloud platforms. To solve these problems, researchers focus on auditing privileged users' behaviors. However, it is difficult to automatically audit fine-grained privileged behaviors for graphical operating systems. Moreover, it is hard to prevent users from bypassing the audit system or to prevent hackers from attacking audit system. In this paper, we propose a Secure and Automatic Behavior Audit system named SA-UBA. It provides advanced deep learning models to automatically achieve fine-grained user behavior audits for graphical operating systems. Furthermore, it adopts cryptography-based account storage and sharing methods to securely manage privileged accounts. In particular, privileged accounts cannot be leaked even if SA-UBA is compromised by attackers. We built a threat model of a cloud platform to evaluate the security of the SA-UBA and conduct extensive experiments with SA-UBA in real scenarios. The results show SA-UBA introduces a small overhead on securely managing privileged accounts and accurately recognizes fine-grained user behaviors.

Keywords: user behavior audit; privileged account management; cloud security

check for updates

Citation: Pan, H.; Han, P.; Xiang, X.; Duan, S.; Liu, C. SA-UBA: Automatically Privileged User Behavior Auditing for Cloud Platforms with Securely Accounts Management. *Appl. Sci.* **2022**, *12*, 8763. https://doi.org/10.3390/app12178763

Academic Editors: Howon Kim and Thi-Thu-Huong Le

Received: 30 June 2022
Accepted: 26 August 2022
Published: 31 August 2022

1. Introduction

Cloud computing allows users and applications to access various services from remote locations, which brings great convenience. Besides ordinary users interacting with cloud services, administrators or management applications with privileged accounts often perform privileged operations to achieve specific tasks [1], such as deleting virtual hosts and downloading database files.

As privileged accounts have much higher permissions than ordinary accounts, they may incur severe security problems on cloud platforms. In particular, when the behavior of a privileged account is out of control, important resources and private data on cloud platforms can be accessed by malicious attackers. For example, millions of records of Uber's personal information about its passengers and drivers were disclosed since the hackers obtained the Uber employees' login credentials from the Amazon Web service [2]. Moreover, even if privileged accounts are not leaked, insiders with privileged accounts can perform dangerous operations that may lead to privacy data leaks or even server crashes [3]. According to the investigation of the CERT Insider Threat Center of Carnegie Mellon University [4], more than 90% of security incidents come from insiders. Therefore, it is important to securely manage privileged accounts and audit privileged user behaviors.

Privileged access and uncontrolled privileged accounts may pose three threats to the cloud platform: firstly, privileged users may perform hidden dangerous operations on the cloud platform that are difficult to identify accurately by the audit system; secondly, privileged users directly access the cloud platform bypassing the audit system when managing the privileged accounts themselves; finally, in the process of users accessing the cloud platform, attackers can invade the system and steal the password by parsing the application code and retrieving hard-coded credentials. To address the above security threats, several studies are auditing privileged user behaviors [5–8] to prevent dangerous operations. They analyze privileged behaviors from RDP (Remote Desktop Protocol) videos [5,6] or network traffic [7,8]. However, it is difficult to automatically audit fine-grained privileged behaviors for graphical operating systems. Moreover, to prevent users from bypassing the audit system or tampering with the behavior logs, various privileged account management systems are proposed [9–13]. They are responsible for managing the remote desktop access of privileged accounts [10], privileged identities in enterprises [9], and database credentials for privileged accounts [11–13]. Although they achieve diversified management functionalities, they fail to consider preventing the leakage of privileged accounts when the privileged account management systems are compromised.

In this paper, we propose a Securely and Automatically Privileged User Behavior Auditing for Cloud Platforms, named SA-UBA. It can automatically audit fine-grained user behaviors in graphical operating systems and securely manage privileged accounts. SA-UBA can audit privileged behaviors in a fine-grained way for graphical operating systems. It first collects operating images when the users access cloud resources and issue commands. After that, SA-UBA applies deep-learning-based text reading to extract text information from images. Particularly, we improved the deep learning models on processing text detection and recognition so that the complex background and small icons of graphical operating systems can be well handled. Finally, SA-UBA matches the extracted text information to specific user behaviors. Thus, SA-UBA can automatically achieve a fine-grained user behavior audit for graphical operating systems.

To prevent users from bypassing the audit system or tampering with the behavior logs, SA-UBA introduces the account-password management module. It conducts resource access authentication for privileged accounts and achieves unified management. Moreover, cryptography-based account storage and sharing methods are proposed to prevent password leakage, even if the system is compromised. As attackers may also try to read authentication credentials in the system or application codes, this system adopts hard-coded elimination and code obfuscation technology to securely store all of the credentials.

We implemented the prototype of SA-UBA and conducted extensive experiments with SA-UBA in real scenarios. Our experimental results demonstrate that SA-UBA introduces a small overhead. For example, it incurs less than 6% band-width usage and about 5% time usage for users with privileged accounts to access cloud resources. Moreover, SA-UBA can detect various texts from complex images with a 22% improvement compared to existing work, and achieve a 94% accuracy in extracting text information. Based on the extracted information, SA-UBA can recognize fine-grained user behaviors with 75% accuracy.

In summary, our paper makes the following contributions:

1. We propose a system named SA-UBA to provide automated and fine-grained user behavior audit method for cloud platforms, especially for graphical operating systems;
2. To prevent users from bypassing the audit system or tampering with behavior logs, we propose a secure privileged accounts management method with a cryptography mechanism;
3. We implement the prototype of SA-UBA and conduct extensive experiments in real scenarios to demonstrate the effectiveness of automated auditing and analyze the security of SA-UBA. We also analyze in detail the introduction of overhead as a supplementary experiment.

The rest of this paper is organized as follows. Section 2 reviews the related studies and compares SA-UBA with existing solutions. Section 3 introduces the overall structure of the SA-UBA. Sections 4 and 5 present the detailed design of this system. Section 6 evaluates the overhead and effectiveness of the system. Section 7 concludes our paper.

2. Related Work and Comparison

Dinoor et al. [14] pointed out that companies using cloud services should take measures to prevent the abuse of privileged access. Sindiren E et al. [15] conducted a detailed analysis of cloud platform security incidents, and their views showed that the lack of privileged account management and user behavior audits were the culprits of these problems. Moreover, Walker P [16] analyzed the reasons for the failure of certain privileged access management systems, and pointed out that a privileged access management system should include two functions: privileged account management and user behavior audits. In recent years, many researchers have started their work from these two aspects.

User Behavior Audit. A number of studies [5–8] prevented dangerous operations by auditing privileged user behaviors. They mainly captured videos and manually audited user behaviors. Zhang et al. [5] designed an audit system based on the RDP proxy, which aimed to prevent malicious users from tampering with audit information. Cui et al. [6] also provided a similar audit system with the proxy-based structure. As manual audits find it difficult to handle large-scale privileged user operations, researchers have applied machine learning models to mitigate the heavy task of large-scale behavior audit. Dusi M et al. [7] first used statistical classification to analyze the network traffic that is generated from user behaviors. It can determine what applications are used during the user's access. On this basis, Suznjevic et al. [8] adopted decision trees to analyze the network traffic and improve the classification accuracy. S. Khaliq et al. [17] collected and analyzed the user's operation logs (such as system logs, application logs, etc.) to obtain ongoing user activities. Junfeng Qi and Yao Y [18,19] used sense text reading to extract the text information from the video. However, existing studies cannot easily automatically audit the fine-grained privileged behaviors for graphical operating systems. In contrast, our system can automatically recognize fine-grained user behavior audit for graphical operating systems.

The User Behavior Audit subsystem designed in this paper can audit privileged behaviors in a fine-grained way for graphical operating systems. We applied deep-learning-based text reading to extract text information from images. In particular, we improved the deep learning models in processing text detection and recognition, so that the complex background and small icons of graphical operating systems can be well handled. Finally, the SA-UBA matches extracted text information to specific user behaviors. Thus, SA-UBA can automatically achieve a fine-grained user behavior audit for graphical operating systems.

Privileged Account Management. To mitigate the security problems caused by privileged accounts, a number of privileged account management systems are proposed [9–13]. Bhaskaran K [9] designed a system deployed on the user side for managing the remote desktop access of privileged accounts. Tan Z et al. [10] provided a proxy-based privileged account management system to manage privileged identities in enterprises. Secure Proxied Database Connectivity (SPDC) [11,12] is a security access system for managing database accounts on the cloud. It can realize the secure connection from applications to cloud-storage data. Sade Y et al. [13] employed the database connection API when the application requested to access the database. W. Tirtadjaja et al. [20] designed a package solution combining privileged access management, and identity and access management. However, no specific technical details were provided. The existing solutions achieve diversified management functionalities. However, they fail to consider preventing the leakage of privileged accounts when privileged account management systems are compromised.

Our system named SA-UBA applies several techniques to enhance the security of managing privileged accounts. We show the comparisons between SA-UBA and existing systems in Table 1. SA-UBA enables resource access authentication for privileged accounts and achieves unified management. Moreover, the cryptography-based account storage and sharing methods are applied to prevent password leakage even if the system is compromised. As attackers can possibly try to read authentication credentials in the system or application codes, SA-UBA adopts hard-coded elimination and code obfuscation technology to securely store all of the credentials.

Table 1. Comparison of privileged account management and user behavior audit.

Related Work	Objects Managed	Client Agent	Modify Code	Code Obfuscation
[9]	User	√	—	—
[10]	User	×	—	—
[20]	User	×	—	—
[12]	Application	√	√	×
[13]	Application	×	×	×
Our System	Both	×	×	√

Related Work	Video Audit	Fine-Grained	Automatically Audit
[5–8,17]	×	×	√
[18,19]	√	×	×
Our System	√	√	√

3. Structure of SA-UBA

SA-UBA consists of two subsystems. The User Behavior Audit subsystem is used to determine the behaviors of privileged users in accessing resources, monitoring these behaviors during the event, and auditing them afterward. We propose a new deep-learning-based text information extraction module for the user behavior audit applied in this subsystem, tackling the challenge of the automatically graphical system audit. The Privileged Account Management subsystem is used to manage privileged accounts by providing secure privileged account storage and management, providing the security guarantee of a user behavior audit. To ensure the security of the system itself, the subsystem adopts cryptography-based account storage and sharing methods to prevent password leakage even if the system is compromised. Hard-coded elimination and code-obfuscation technology have also been adopted to protect the codes. As shown in Figure 1, SA-UBA implements three functions: the red line represents the user-access process, the blue line represents the application-access process, and the yellow line represents the user behavior audit process.

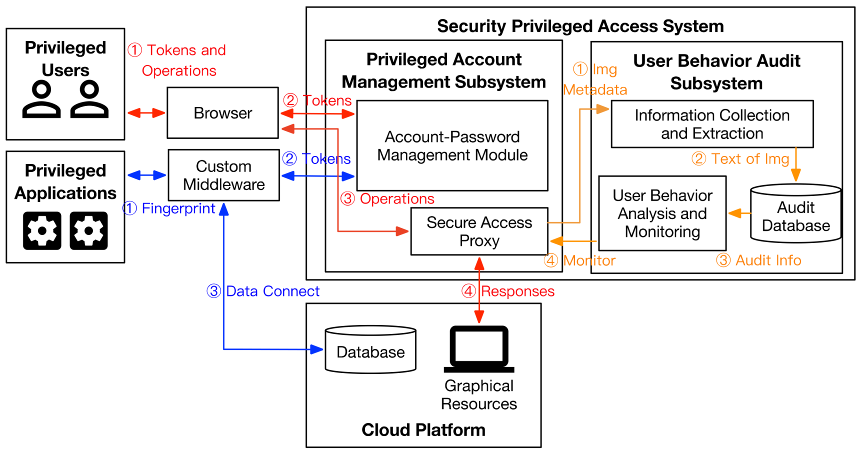


Figure 1. Overall structure of SA-UBA. The red line represents the user-access process, the blue line represents the application-access process, and the yellow line represents the user behavior audit process. Before users and applications access the system, all privileged accounts will be statistically stored and managed by the Accounts Password Management module.

- **Secure access to the cloud resources by privileged users:** As shown by the red line in Figure 1: (1) The SA-UBA system allows users to access cloud platform resources with a browser. (2) Before establishing the connection with a resource on the cloud, the users' identity needs to be authenticated by the Accounts Password Management module. If the authentication succeeds, the token will be returned to the user for accessing the resource. (3) The user applies to the Secure Access Proxy with the token to access resources. Once the connection is established, the proxy is responsible for communicating between the users and resources. (4) The Secure Access Proxy maintains the connection between users and resources. This proxy can help to extract audit information and monitor user behavior based on the analysis results of the User Behavior Audit Subsystem.

- **Secure access to the cloud database by privileged applications:** As shown by the blue line in Figure 1: (1) Custom Middleware is designed to assist in the authentication of applications. Therefore, the application needs to ask the Custom Middleware for access to the cloud resources. (2) The Accounts Password Management module authenticates the fingerprint of the application and returns a token if successful. (3) Custom Middleware establishes a connection, using the token, between the certified applications and the database.
- **User behavior audit:** As shown by the yellow line in Figure 1: (1) Firstly, the user operation images and operation-related metadata are passed to the User Behavior Audit subsystem; these data are generated in the Secure Access Proxy and collected by the User Behavior Collection module. (2) The Text Information Extraction module extracts the text information in the user operation images and outputs the audit data to the Audit Database. The deep-learning-based scene text-reading model is used in this step to obtain more accurate audit data. (3) The User Behavior Analysis module identifies the users' operations by analyzing the audit data. This module divides user behavior into four levels to accurately determine privileged operations and respond in time. (4) The result of the user behavior analysis is sent to the Secure Access Proxy. When the user is found to perform an illegal operation, the proxy will block the operation in time.

4. Privileged User Behavior Audit Subsystem

This section gives the design of the Privileged User Behavior Audit subsystem. To solve the problem that it is difficult to supervise the behavior of privileged users, this subsystem provides automated auditing and monitoring functions. The structure of this subsystem is shown in Figure 2.

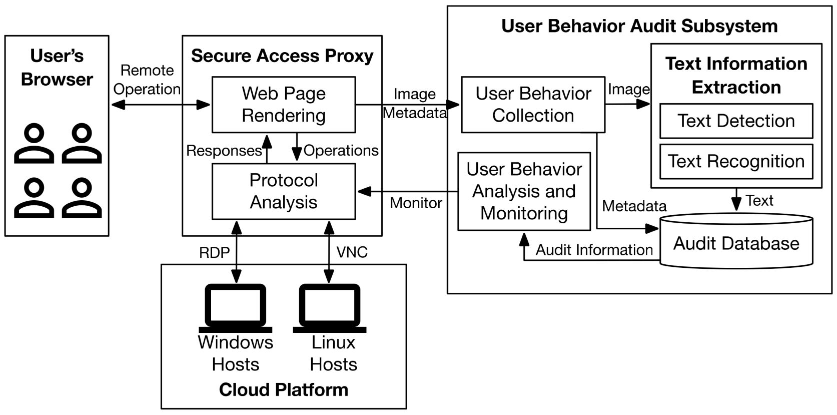


Figure 2. The structure of privileged user behavior audit subsystem.

Users access the cloud resources through the secure access proxy. The web page's drawing function receives drawing instructions that carry a lot of audit information (including images and metadata). The User behavior image collection module (Section 4.1) collects and parses these instructions, restoring images and extracting metadata. For these images, the text information extraction module (Section 4.2) locates the text area and recognizes the text content, the core of this module is scene text-reading based on deep learning. The text information extracted from the images and the metadata are stored in the audit database as audit information, the user behavior analysis and monitoring module (Section 4.3) determines the users' behaviors and intercepts against actions outside the users' authority.

4.1. User Behavior Logs Collection

The User Behavior Collection module is designed to extract the user operation images and operation-related metadata when users access the cloud platform resources. These data are contained in "img" Instruction, "blob" Instruction and Connection Information.

The key arguments are shown in Table 2. Among these arguments, "userID", "userIP", and "serverIP" come from the connection information, "opTime", "imgX", and "imgY" come from "img" Instructions and "imgCode" comes from the "blob" Instructions. All of these arguments will be stored in the Audit Database as one piece of audit data.

Table 2. Arguments of audit data.

Argument Name	Argument Meaning	Examples
userID	A string that uniquely identifies the user.	ID56127848
userIP	User's IP address.	192.168.0.101
serverIP	IP address of Resource.	192.168.0.105
opTime	The time the user took the action.	2020-03-05 18:10:12
imgX	The X coordinate of the upper-left corner.	(2.14, 1.09)
imgY	The Y coordinate of the upper-left corner.	(3.475, 2.32)
imgCode	The base64-encoded data of the image.	iVBORw0KCYII=
text	Text information on the image.	My Computer

"imgCode" is the base64-encoded data of images, these images will be decoded and transmitted to the Text Information Extraction module. The argument "text" is the output of the Text Information Extraction module, including all of the text information on the image. The generation process of Argument "text" will be described in the next section.

4.2. Deep-Learning-Based Text Information Extraction

To extract the text information in these images, deep-learning-based scene text-reading is used in this module. In this part, we draw lessons from the related work of scene text-reading and modified for the user behavior audit scenario. The Text Information Extraction module can be divided into the text detection process and text recognition process, as is shown in Figure 3.

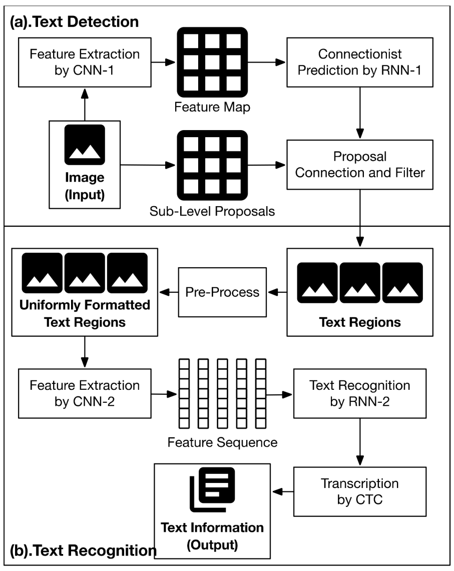

(a).Text Detection
Feature Extraction by CNN-1
Feature Map
Connectionist Prediction by RNN-1
Image (Input)
Sub-Level Proposals
Proposal Connection and Filter
Uniformly Formatted Text Regions
Pre-Process
Text Regions
Feature Extraction by CNN-2
Feature Sequence
Text Recognition by RNN-2
Transcription by CTC
Text Information (Output)
(b).Text Recognition

Figure 3. Text information extraction module structure based on deep learning.

4.2.1. Text Detection Process

The text detection process in this module aims to detect the text regions from the images representing user behaviors. The images generated during user access to cloud computing resources usually contain horizontal text. The deep-learning-based text detection model Connectionist Text Proposal Network (CTPN) [21,22] happens to be suitable for this situation. Aiming at the problem of a complex background in the user behavior audit scene, the CTPN model is specifically modified in this paper. The process is shown in Figure 3a: 1. The image input to the Text Information Extraction module will be divided into multiple sub-level proposals. The main idea of the CTPN model is to determine whether each proposal contains text, and which connect proposals are most likely to contain texts. 2. At the same time, the image will be input into the first Convolutional Neural Network (CNN) to extract the feature map. 3. Considering the sequence and locality of the text, the Recurrent Neural Network (RNN) is used for connectionist prediction on the feature map. To fully analyze the relationship between the adjacent proposals, the Bi-directional Long Short-Term Memory network (BiLSTM) is used to replace traditional RNN. 4. According to the prediction for each proposal given by BiLSTM, the text detection model will find out the proposals containing texts and connect the adjacent proposals into text regions according to the algorithm proposed in [21].

Targeted modification of CTPN: During the text detection process in the user behavior audit scene, it is difficult for the CTPN model to distinguish between the text-like graphics (such as small icons in the Windows system) and text areas. In this article, we tried to solve this problem by modifying the predicted value of the RNN model. The original CTPN model predicts five variables for each proposal. Among these variables, the text and non-text scores are used to determine whether a proposal contains text and the other three variables are used to adjust this proposal to make it more likely to contain text. This adjustment will only change the size and vertical position of the proposals, not the horizontal position. This design may cause the text-like graphics to be mistakenly framed into text regions when the width of the proposals increases. The modified model predicts four different variables for the adjustment of proposals, so that as the width of a proposal increases, its horizontal position will also change. The mathematical expression of modified CTPN is shown as follows:

$$t_x = \frac{G_x - A_x}{A_w},\ \ t_x^* = \frac{G_x^* - A_x}{A_w} \tag{1}$$

$$t_y = \frac{G_y - A_y}{A_h},\ \ t_y^* = \frac{G_y^* - A_y}{A_h} \tag{2}$$

$$t_w = log(G_w/A_w),\ \ t_w^* = log(G_w^*/A_w) \tag{3}$$

$$t_h = log(G_h/A_h),\ \ t_h^* = log(G_h^*/A_h) \tag{4}$$

where T = (t_x, t_y, t_w, t_h) and T* = $(t_x^*, t_y^*, t_w^*, t_h^*)$ are the ground truth coordinates and the predicted coordinates, respectively. G = (G_x, G_y, G_w, G_h) and G* = $(G_x^*, G_y^*, G_w^*, G_h^*)$ are the ground truth box and the predicted box, respectively. G_x,G_y,G_w,G_h are the *x*-axis, *y*-axis of the center and the width, height of the ground truth text box, respectively. A_x, A_y, A_w, and A_h are the *x*-axis, *y*-axis of the center, and the width, height of the text proposal.

The overall objective function (L) contains two loss functions L_{cls} and L_{reg}, which compute errors of text/non-text scores and proposal coordinates, respectively. We follow the multi-task loss applied in [21] to minimize the objective function (L) for an image as:

$$L\left(s_i, s_i^*, t_j, t_j^*\right) = \frac{1}{N_{cls}} \sum_{i}^{N_{cls}} L_{cls}(s_i, s_i^*) + \frac{\lambda}{N_{reg}} \sum_{j=1}^{N_{reg}} L_{reg}\left(t_j, t_j^*\right) \tag{5}$$

$L_{cls}(s_i, s_i^*)$ and $L_{reg}\left(t_j, t_j^*\right)$ are as follow:

$$L_{cls}(s_i, s_i^*) = -log(s_i s_i^* + (1 - s_i)(1 - s_i^*)) \tag{6}$$

$$L_{reg}\left(t_j, t_j^*\right) = smooth_{L_1}\left(t_j - t_j^*\right) \tag{7}$$

where $smooth_{L_1}$:

$$smooth_{L_1}(x) = \begin{cases} 0.5x^2 & if|x| < 1 \\ |x| - 0.5 & otherwise \end{cases} \tag{8}$$

where s_i^* is the predicted probability of proposal i being a true text. s_i = {0, 1} is the ground truth. N_{cls} and N_{reg} are the number of proposals used by L_{cls} and L_{reg}, respectively. λ is the loss weight to balance different tasks; in this paper, we set it to 2.0. The effect of this modification will be evaluated in Section 6.

4.2.2. Text Recognition Process

The text recognition process aims to transform the text regions into editable characters (the argument "text" in the audit database). Considering that the text images belong to sequence-like objects, the Convolutional Recurrent Neural Network (CRNN) [23,24], a combination of CNN and RNN, is adopted in this paper. The images generated during user operations often contain incomplete characters. Therefore, a pre-process is designed to segment the text regions and remove incomplete characters.

Pre-process before text recognition: The pre-process modifies the text regions to uniformly formatted text regions of the same size and no incomplete characters. This process mainly contains three steps: 1. Segmenting the text regions based on the OSTU [25] algorithm, the segmentation in this step is according to the gap between characters; 2. Removing text regions whose size is below the threshold, these small text regions are considered incomplete words or characters; 3. Scaling all of the remaining text areas to a fixed size through the bi-linear sampling algorithm. The unified formatting of the text regions improves the accuracy of text recognition. This conclusion will be explained in Section 6.

As is shown in Figure 3b: 1. A text region is converted to feature sequence by CNN-2, this CNN model is specially designed to divide the text region vertically into several parts, and each part corresponds to a feature sequence. 2. RNN-2 analyzes these sequences and predicts the characters to which each sequence may correspond, the output of RNN-2 is that each sequence corresponds to any character. 3. Connectionist Temporal Classification (CTC) [26] is used to transform the probability predicted by RNN-2 into the final character sequence.

For instance, CNN-2 divides the text region into 12 parts with 12 feature sequences. RNN-2 predicts the characters to which these 12 parts may correspond, and the output result is the sequence "-u-ss-er-ID-" ("-" means RNN-2 predicts this part as blank). CTC transforms this sequence into "userID" by computing the probability.

4.3. User Behavior Analysis Monitoring

In the User Behavior Analysis Monitoring module, the text of the image—the result of the deep-learning-based Text Information Extraction—will be matched to the existing keywords in the audit database and determine the users' behavior. This module identifies users' operations by matching the keyword string of the argument "text" in the audit database. For instance, in the audit database, the string "Upload-WinSCP" appears, representing a privileged user uploading files to the remote host, and this operation can be a Privileged Operation.

In this module, users' operations on cloud computing resources will be divided into four categories:

1. **Regular Operation** refers to general operations performed by users when they access cloud computing resources, and such operations will be executed normally;

2. **Privileged Operation** involves key resources or sensitive data on the cloud platform. The user behavior monitoring module will block the operation and request approval from the system administrator. Privileged operations can only be performed with administrator approval;
3. **Unauthorized Operation** may affect resources or data that do not belong to the user, the user behavior monitoring module will prevent such operations directly;
4. **Dangerous Operation** is destructive to resources and data on the cloud platform, such as deleting databases and deleting virtual machine images. This module will disconnect the user from the cloud platform while preventing such operations.

All of the user behaviors will correspond to the above four types of operations, and the User Behavior Monitoring module will react based on the classification results. The Audit Database will record three types of operations, Privileged Operations, Unauthorized Operations, and Dangerous Operations, for post audit.

Figure 4 is an actual case to illustrate the process of the privileged user behavior audit. In this case, a privileged user tried to dump the database file and send it to the remote host. During the user operation, the User Behavior Collection module collected operation-related metadata (Figure 4a,b) and five images (Figure 4c). The Text Information Extraction module extracted the text information in these images and obtained the result shown in Figure 4d. Finally, the User Behavior Analysis module matched the key string in the text information (red words in Figure 4d) and determined the user behavior.

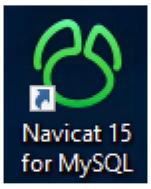
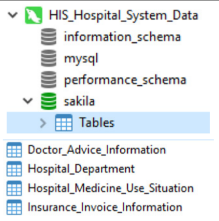
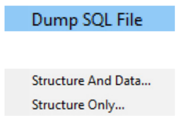
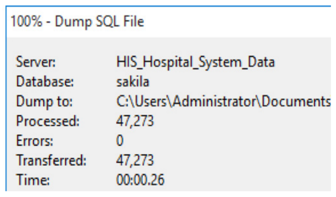
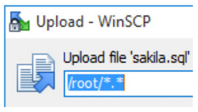
(a).Operation-Related Metadata userID: ID56127848 userIP: 192.168.0.101 serverIP: 192.168.0.105					
(b).opTime	2020.3.1-11:57:32	2020.3.1-12:07:01	2020.3.1-12:07:52	2020.3.1-12:09:18	2020.3.1-12:12:27
(c).Image	 Navicat 15 for MySQL	 HIS_Hospital_System_Data information_schema mysql performance_schema sakila Tables Doctor_Advice_Information Hospital_Department Hospital_Medicine_Use_Situation Insurance_Invoice_Information	 Dump SQL File Structure And Data... Structure Only...	 100% - Dump SQL File Server: HIS_Hospital_System_Data Database: sakila Dump to: C:\Users\Administrator\Documents Processed: 47,273 Errors: 0 Transferred: 47,273 Time: 00:00.26	 Upload - WinSCP Upload file 'sakila.sql' /root/*.*
(d).Text on Image	Navicat 15 for MySQL	HIS_Hospital_System_Data … sakila Table Doctor_Adivce_Information …	Dump SQL File Structure And Data… Structure Only…	100% - Dump SQL File Server: HIS_Hospital_System_Data Database: sakila … …	Upload - WinSCP Upload file 'sakila.sql' /root/*.*
(e).User's Behavior	Access MySQL with Navicat	Access the database of HIS Hospital	Dump SQL File	Dump SQL file completed	Upload File with WinSCP

Figure 4. An actual case of privileged user behavior audit. A privileged user tried to dump the database file and send it to the remote host.

5. Privileged Account Management Subsystem

This section gives a minute description of the Privileged Account Management subsystem. In Section 5.1, the working process of the Account-Password Management module is introduced, which is the core of this subsystem. Based on this module, the User-Access Process and Application-Access Process with Privileged Account Management are elaborated in Sections 5.2 and 5.3.

5.1. Account-Password Management Module

The Account-Password Management module is the core module of the Privileged Account Management subsystem. The users and applications will authenticate to this module before accessing the resources. In this part, we use a cryptography mechanism to store and manage the accounts. Once the system itself is compromised, the cryptography-based

Account-Password Management module can effectively prevent hackers from stealing privileged accounts.

5.1.1. Encrypted Storage

In this subsystem, the account passwords are stored with hierarchical encryption. The encrypted storage scheme is shown in Figure 5. In this scheme, each account password is encrypted with a Data-Encryption Key (DEK), for each application or cloud resource; additionally, all of the DEKs are encrypted with a Key-Encryption Key (KEK), and all of the KEKs are encrypted with the Root Key (RK). AES-256-GCM is used for all of the encryption processes in this part. The secret sharing algorithm [27] can divide the RK into N key blocks, and send them to N administrators of storage. The storage of the account passwords can be decrypted when at least K of N administrators provide their key blocks simultaneously. The secret sharing algorithm is implemented as follows:

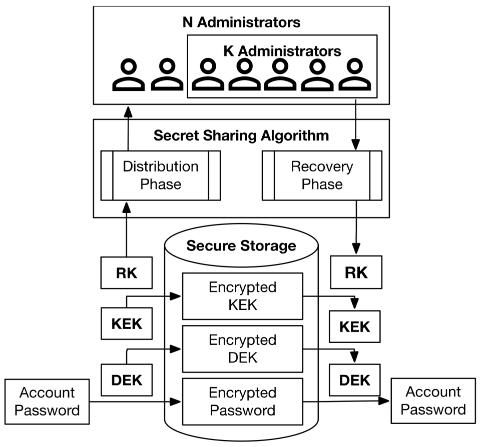


Figure 5. Hierarchical encryption of account passwords. Secret Sharing Algorithm divides RK into N key blocks for N administrators.

Initialization Phase. We define finite field GF(p). Here, p is a large prime number. We then randomly select N non-zero elements ($x_1, x_2 \ldots x_N$) in the GF(p) and send them to the related administrator ($A_1, A_2 \ldots A_N$) in public. Element x_r can represent the identity of administrator $\mathrm{A_r}$.

RK Distribution Phase. We select an integer s less than p as RK, and select (K − 1) elements in the GF(p) as $a_1, a_2 \ldots a_\mathrm{K}$. Equation (9) is a polynomial function where a_0 = s:

$$f(x) = a_0 + \cdots + a_{K-1}x^{K-1} \quad (mod\ p) \tag{9}$$

For each A_r in ($A_1, A_2 \ldots A_\mathrm{N}$), we generate a key blocks s_r by Equation (10):

$$s_r = f(x_r) = a_0 + \cdots + a_{K-1}x_r^{K-1} \ (mod\ p) \tag{10}$$

Then, we send key blocks s_r to A_r in secret.

RK Recovery Phase. Any K administrators ($A_1, A_2 \ldots A_K$) can recover RK by function (3), the Lagrange interpolation formula is used in Equation (11):

$$s = f(0) = \sum_{i=1}^{K} f(x_i) \prod_{j=1, j\neq i}^{K} \frac{x_j}{x_j - x_i} \ (mod\ p) \tag{11}$$

5.1.2. Account-Password Management

After the initialization of the account-password management module, all of the account passwords are encrypted and stored in secure storage. When more than K administrators submit key blocks and recover the RK, the module will decrypt the KEK, DEK, and account passwords in turn and start working. This module contains five functions:

1. **Account Management.** Administrators can register accounts of cloud resources to this module including cloud hosting accounts, database accounts, and other accounts. These accounts will be managed automatically;
2. **Dynamic Password Management.** Connecting to cloud resources hosted by this module, and creating dynamic account password for these resources;
3. **Authentication.** Authenticating privileged users and applications, allowing authenticated users or programs to retrieve their account passwords;
4. **Lifecycle Management.** If a resource has passed the lease term, the module will automatically restore its account password;
5. **Password Rotation.** If the password has not been changed for a long time or is leaked (password change cycle is predefined), this module will automatically change the password; if the storage is attacked, or storage device failure occurs, the account-password management module will be out of work and re-encrypt the KEK, DEK, and account passwords.

5.2. User-Access Process

After administrators registering accounts of cloud resources to the account-password management module, users can access these resources through the privileged account subsystem. The specific process is shown in Figure 6: (1) User authenticates to the account-password management module at first; (2) The authentication function confirms the user's identity and returns a token corresponding to the identity, this token is the user's credentials for accessing resources; (3) The user connects to the account-password management module again for requesting the password of the resource account, the request should contain the user's token; (4) The password management checks the token and returns the account password requested by the user; (5) With this password, the user can access the cloud resource through the secure access proxy.

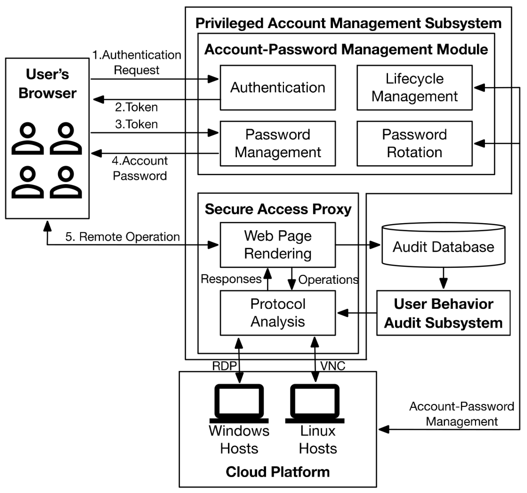


Figure 6. User-access process of Privileged Account Management subsystem.

Secure access proxy is a key module in the User-Access Process. This proxy allows users to access cloud resources with a browser and automatically collects audit images for the user behavior audit subsystem. This proxy can analyze protocols for accessing cloud resources and draw the desktops on the users' browsers.

The protocol analysis function connects to the cloud resources with RDP (Windows system) or VNC (Linux) and translates the requests of these protocols into instructions. There are five commonly used instructions in this function: Handshake instructions are used to establish a connection; Control instructions are used to control the connection such as “close” and “reset”; Drawing instructions can be executed through the webpage drawing function, used to draw a remote desktop on the user’s web browser; Client events instructions are used to transmit users’ keyboard or mouse input events; Streaming instructions are used to transfer binary objects, such as pictures, videos, files, etc. Each instruction consists of an opcode and several arguments; the opcode and arguments are used to indicate the operation type and operation details.

The web page drawing function receives drawing instructions from the protocol analysis function and draws the remote desktop on users’ browsers in real-time. Drawing instructions are only generated when the remote desktop changes, so the web page drawing function collects details of these instructions for user behavior auditing. For instance, “img” instructions and “blob” instructions are transmitted to the web page drawing function, “img” instruction contains information of an image for drawing the remote desktop, and “blob” instruction contains the base64 encoding of the image. For auditing the user’s behavior, the image in “blob” instruction and metadata information of these two instructions are recorded and stored in the audit database.

5.3. Application-Access Process

The application-access process is the interaction between the applications and databases on the cloud platform. As with the user access process, administrators need to register accounts of the database and fingerprint of applications at first. The specific process is shown in Figure 7: (1) When an application initializes the connection to a database on the cloud platform, Custom Middleware extracts the fingerprint of this application; (2) Custom Middleware submits the fingerprint to the account-password management module to authenticate the application; (3) The authentication function verifies the fingerprint and returns the token of this application; (4) Custom Middleware requests the password of the database with the token; (5) The password management checks the token and returns the password; (6) Custom Middleware establishes a connection between the application and the database with the database password.

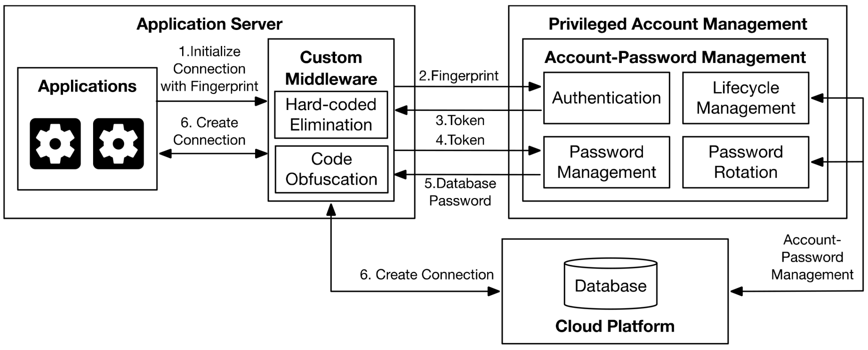


Figure 7. Application-access process of Privileged Account Management subsystem.

Custom Middleware is deployed on the application server and is the key module. In the application-access process, Custom Middleware is used to authenticate applications’ identity with the account-password management module and to establish a connection between the application and database. In the design of Custom Middleware, we considered two aspects of security: first, once the application code is leaked, the hacker can extract the authentication credentials stored in the clear text in the application; secondly, when the SA-UBA is attacked, hackers can reverse analyze the custom middleware code and forge authentication requests. Therefore, this paper adopts the following two defense methods:

Hard-coded Elimination. The connection between the application and the database is usually executed through the database-driver. To establish a connection by database-driver, an application needs to store their credentials and submits the credentials to the database for initializing the connection. Once the credential is revealed during the communication, the attacker can use it to access the database and steal data. To solve this security issue, the hard-coded elimination function rewrites the database-driver replacing the credential with the dynamic password generated by the account-password management module. If the password is leaked, the password rotation function can change the password in time to prevent the attackers from accessing the database.

Code Obfuscation. If the attackers set reverse engineering on to the custom middleware, authentication and password retrieval requests may be forged. To prevent this attack, code obfuscation is used with the database-driver. The code obfuscation method used in this article is as follows [28]: 1. Layout Transformations: Removing comments and formatting and jumbling the identifiers; 2. Control Flow Transformations: Transforming the control flow of the database-driven program. Execution includes aggregating uncorrelated computations and creating randomization in the order of expressions, statements, and loops. 3. Data Abstraction Transformations: Changing the name, inheritance relations, or the structure of data arrays to obfuscate data abstractions.

6. Evaluation

This section is an overall evaluation of the whole system. First, we built a threat model of a cloud platform to evaluate the overall security of the SA-UBA. After that, the Automatic User Behavior Auditing function was tested in two aspects: (1) We focused on testing the improvement effect of a deep-learning-based Text Information Extraction module, compared with the original model; (2) The overall performance of this subsystem was tested by simulated user operations. At last, the overhead introduced by this system was also tested in two aspects: (1) For the user-access process, we mainly evaluated the additional network overhead and resource occupation caused by secure access; (2) For the application-access process, this article took "JAVA Application to Database" as an example to test the delay caused by the security mechanism in three different scenarios. Table 3 is the checklist of using data for the evaluation part. This table shows the size, format, processing, and use of the data set in the evaluation part.

Table 3. Checklist of using data (Evaluation of User Behavior Auditing Function).

Data Set	Data Size	Data From	Data Process	Uses
user operation images scenario 1	200	Images and MetaData	Data Acquisition, A unified image size, Annotation texts	Evaluation of Text Information Extraction module
user operation images scenario 2	200	Images and MetaData		
user operation images scenario 3	200	Images and MetaData		
privileged behaviors	300	Video and labels	Data Acquisition, Labeling	Evaluation of user behavior analysis

As shown in Figure 1, the Security Privileged Access System was deployed on a server independent of the cloud platform with Intel Core i5 dual-core CPU and 4.0 GB RAM. The User Behavior Audit Subsystem was deployed on another server with Intel Xeon four core CPU and 8.0 GB RAM. Although the training of the deep-learning model ran on a GPU server, a CPU server was used in the real running environment in this evaluation. The network settings in this experiment were the downstream bandwidth of 8192 Kbps and the upstream bandwidth of 4096 Kbps.

6.1. Threat Model and Security Evaluation

Privileged accessing without auditing and monitoring may incur severe security problems on cloud platforms. The threats of the privileged accessing process in cloud platform can be summed up in three points:

- **Threat 1:** As is shown in Figure 8 red line No. 1, privileged users or applications may perform hidden dangerous operations on the cloud platform, this kind of operation usually consists of a series of complex operations and is difficult to identify accurately by the audit system. User behavior auditing can be more difficult for graphical operating system. Solution 1: In this paper, we provide automated and fine-grained user behavior audit method. This method can find hidden dangerous operations effectively, especially for graphical operating systems;

- **Threat 2:** As is shown in Figure 8 red line No. 2, if users or applications manage the passwords of privileged accounts themselves, they can directly access the cloud platform bypassing the audit system. Solution 2: In this paper, we propose a secure privileged accounts management method, managing these accounts centrally. The passwords of the privileged accounts are stored encrypted by symmetric key cryptography, users or applications can only obtain the password when accessing the cloud platform. These passwords will be reset periodically;
- **Threat 3:** As is shown in Figure 8 red line No. 3, in the process of users or applications accessing the cloud platform, attackers can invade the system and steal the password, attacking may include parsing the application code and retrieving the hard-coded credentials. Solution 3: In this paper, we propose a custom middleware to implement hard-coded elimination and code obfuscation. Two stealing methods of attackers are prevented pertinently by the custom middleware.

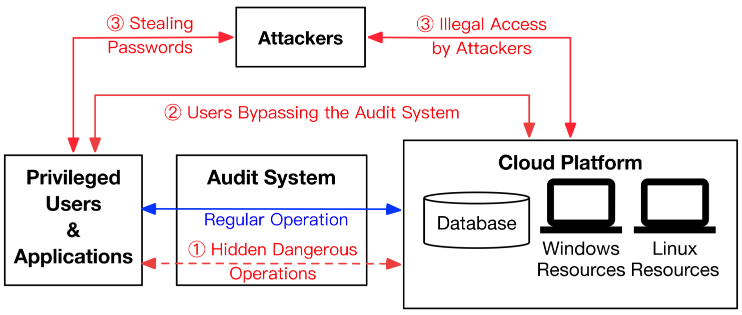

③ Stealing Passwords
Attackers
③ Illegal Access by Attackers
② Users Bypassing the Audit System
Privileged Users & Applications
Audit System
Regular Operation
① Hidden Dangerous Operations
Cloud Platform
Database
Windows Resources
Linux Resources

Figure 8. Threats of privileged accessing process in cloud platform.

6.2. Evaluation of Automatic User Behavior Auditing Function

6.2.1. Evaluation of Deep-Learning-Based Text Information Extraction Module

To evaluate the Text Information Extraction module in the real privileged user behavior audit, this experiment selected three scenarios to simulate users' operations, (Windows system, applications, and browsers). When simulating users' behaviors, the User Behavior Collection module extracted the user operation images in real time. In this step, a large number of user behavior images were collected. We manually filtered the images that did not contain text information or were too small, and 600 representative images (200 images for each scenario) were kept.

Both the text detection process and text recognition are improved from existing deep-learning-based scene text-reading models. This experience mainly compares the performance of the original model and the improved model in the user behavior audit scene. The DetEval algorithm [29] is used to evaluate the text detection process and the text recognition process is evaluated by the Edit Distance. The comparison results are shown in Figure 9.

The improvement of the text detection model greatly improved the accuracy in the applications' and browsers' scenarios, the F-Score increased by 29.7% in the applications' scenario and 26.7% in the browsers' scenario. The modified model can better handle text-like graphics in the user behavior audit scenario, this ability makes it more accurate in two scenarios with complex backgrounds. On the other hand, the accuracy of the text recognition model improves from 90% to 94%, this improvement supports the analysis of user behavior to a certain extent.

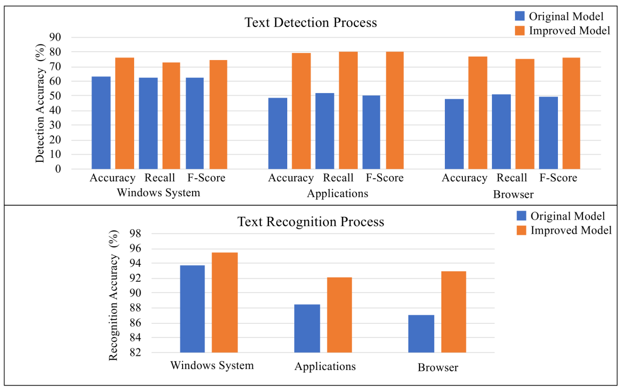


Figure 9. The accuracy of the original model and the improved model.

6.2.2. Evaluation of User Behavior Analysis

In this part, we will evaluate the overall accuracy of user behavior recognition. For this purpose, this paper simulates three types of privileged behaviors in the graphical operating system:

1. **System Operations:** Adding an account or changing a password in Windows system;
2. **File Modifications:** Opening a word processor and modifying a sensitive document;
3. **Resource Access:** Daily operations of operation and maintenance personnel including modifying the configuration file and accessing the database.

These behaviors are simulated by Auto IT and analyzed by the User Behavior Audit subsystem. Figure 4 shows the process of the user behavior audit. The accuracy of user behavior recognition is shown in Table 4. The experimental results show that the overall accuracy of user behavior auditing is about 74%, and the accuracy of identifying complex operations, such as resource accessing, is not less than 70%.

Table 4. Accuracy of user behavior recognition.

Behavior Type	Number of Operations	Correct Recognition
System Operations	100	78
File Modifications	100	73
Resource Access	100	70

Table 5 shows the time cost of user behavior recognition. The time cost of text detection processes and text recognition processes are 1.06 s and 1.53 s on average, and the whole user behavior recognition process will cost 2.59 s on average. In the SA-UBA system, the user behavior audit program runs on an independent server, so it will not affect the operation of the user and the server on the cloud platform.

Table 5. The time cost of user behavior recognition.

Process	Minimum Time	Maximum Time	Average Time
Text detection	0.09 s	2.63 s	1.06 s
Text recognition	0.23 s	9.67 s	1.53 s
Whole process	0.32 s	11.74 s	2.59 s

6.3. Evaluation of Overhead Introduced by SA-UBA

6.3.1. Evaluation of User-Access Process

In the User-Access Process, the network bandwidth overhead is commonly used to evaluate the user's experience of accessing resources on the cloud. This experiment compares the network overhead of direct accessing (connecting resources with mstsc.exe) and accessing through Privileged Account Management subsystems. Three types of user behaviors are used in this test: (1) Edit: User opens a word processor (such as notepad, Microsoft Word, etc.) and enters text; (2) Browse: User visits some web pages through the browser on cloud hosts; (3) Video: The user uses video applications or plays video files. The researchers repeat each operation 10 times and calculate the average of the network bandwidth overhead in 30 s.

The results of this experiment are shown in Figure 10; this figure compares the peak, valley, and average values of the bandwidth overhead when users perform three operations. The peak values gap between direct accessing and accessing with the Privileged Account Management subsystem is within 0.04 Mbps, and the valley values gap is within 0.03 Mbps. These gaps come from factors such as network instability or bandwidth changes. The gap of the mean network bandwidth overhead is not more than 0.06 Mbps. In other words, the additional bandwidth overhead brought by the Privileged Account Management subsystem is less than 5.7%. In practice, this additional overhead will not affect users' normal access to cloud resources.

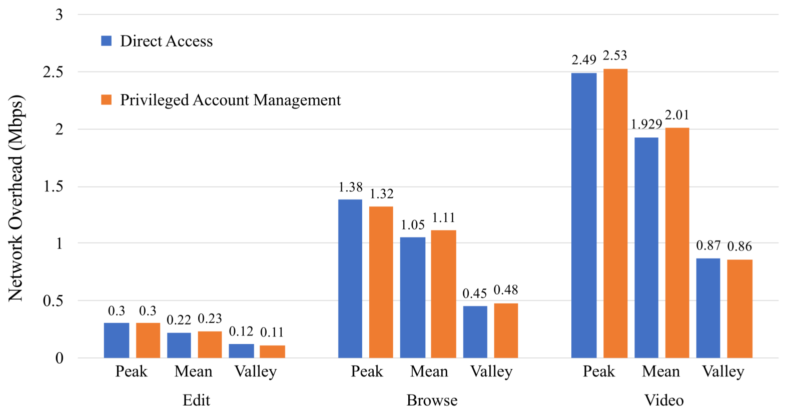


Figure 10. The accuracy of the original model and the improved model.

As a secure access system for cloud platforms, the ability to handle multi-user access requests is crucial. The resource occupation of the Privileged Account Management subsystem is shown in Figure 11. This experiment tested the CPU and memory usage of the secure server during multi-user access. The experimental results show that a server with a dual-core CPU and 4.0 GB RAM can support 50 users to access at the same time. When 20 users access at the same time, the CPU and memory usage is about 60%, which is the normal load of the security server.

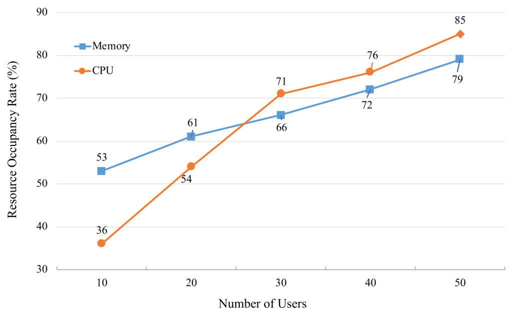


Figure 11. The accuracy of the original model and the improved model.

6.3.2. Evaluation of Application-Access Process

It is a typical application-access process for a Java application to access databases on the cloud platform through the Java database connectivity (JDBC) interface. Generally speaking, there are three authentication scenarios between Java applications and databases on the cloud platform:

- **Scenario 1:** Java program developers directly embed authentication credentials in the source code;
- **Scenario 2:** Authentication credentials are embedded in the configuration file of the Java-based framework (such as SpringBoot, SpringMVC, etc.). The framework reads these authentication credentials and passes them to the database connection pooling (Druid [30], HikariCP, etc.). Database connection pooling creates the database connection and returns the response to the framework;
- **Scenario 3:** Authentication credentials are embedded in the configuration file of a web container. Java web applications (such as Java Servlet) are deployed in web containers (such as WebLogic, WebSphere, JBoss, Tom-cat). Web containers create the database connection and Java web applications use the connection with the Java Naming and Directory Interface (JNDI).

In all of these three scenarios, the Privileged Account Management subsystem is used to replace the original authentication method. This subsystem eliminates the authentication credentials embedded in Java code and configuration files by building Custom Middleware, and realizes the dynamic authentication and management of applications with the Account-Password management module. In this part, we observed the Insertion, Deletion, Modification, and Query operations performed by the Java application on the database and tested the performance cost of the Privileged Account Management subsystem. In this experiment, the Java application was implemented based on JDK1.8, and MySQL5.7 was deployed on the cloud server. The performance costs in these three scenarios are shown in Figure 12, Figure 13, and Figure 14, respectively.

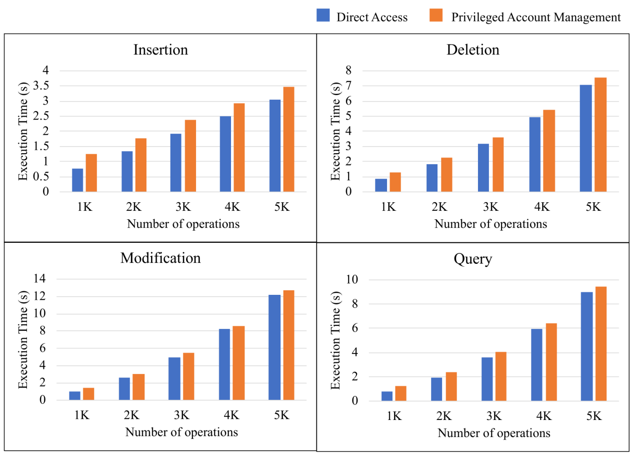


Figure 12. Comparison of execution time between direct access and secure access in scenario 1.

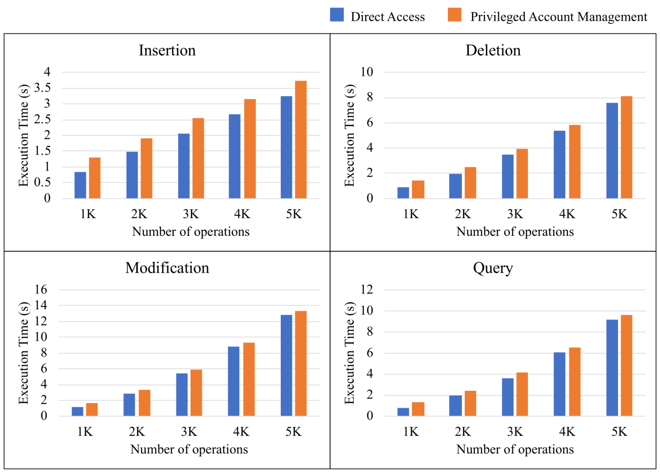


Figure 13. Comparison of execution time between direct access and secure access in scenario 2.

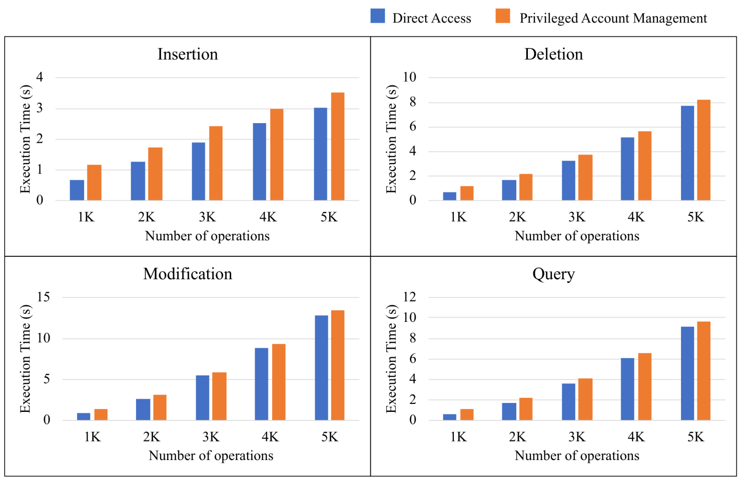


Figure 14. Comparison of execution time between direct access and secure access in scenario 3.

Analyzing the data of the three figures, this paper draws the following conclusions:

1. In these three scenarios, the cost ratios of adding a Privileged Account Management subsystem are 11.89%, 11.82%, and 12.54%, respectively, the cost in the third scenario is the largest but not by more than 13%;
2. When the number of operations increases, the cost ratio will decrease. Taking the Query operation of Scenario One as an example, when the number of operations is 1000, 3000, and 5000, the cost ratios are 59.9%, 12.7%, and 5.0%, respectively. This result occurs because the first authentication between the Java application and the database costs a large amount of time. When performing operations continuously, the time overhead approaches 5%;
3. For Insertion, Deletion, Modification, and Query operations, the increased computing times of the Privileged Account Management subsystem are 24.39%, 12.99%, 8.06%, and 11.34% respectively. The additional time cost of the Insertion operation is the largest. The time cost of security access is about the same (among 0.41 s and 0.53 s). The execution time of the insert operation is short, so the additional overhead takes up a larger proportion.

To sum up, the cost ratios caused by the Privileged Account Management subsystem in the application-access process is not more than 13%. For large-scale and long-term operations, the cost ratio is smaller (about 5.0%).

7. Conclusions

We propose a system named SA-UBA that automatically audits user behaviors for cloud platforms. SA-UBA applies advanced deep learning models to audit privileged behaviors in a fine-grained way for graphical operating systems. In order to strengthen the security of the audit system, SA-UBA applies cryptography-based accounts' storage and sharing methods to securely manage privileged accounts. Even if SA-UBA is compromised by attackers, privileged accounts still cannot be leaked. We implemented the prototype of SA-UBA and evaluated it in real scenarios. The experimental results demonstrate that SA-UBA can securely manage privileged accounts with a small overhead and accurately recognize fine-grained user behaviors.

Author Contributions: Formal analysis, H.P., X.X., and C.L.; Investigation, H.P., P.H., X.X., and S.D.; Validation, P.H. and S.D.; Writing—original draft, H.P. and C.L.; Writing—review and editing, C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (Grant No. 61872110).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Tep, K.S.; Martini, B.; Hunt, R.; Raymond Choo, K.K. A taxonomy of cloud attack consequences and mitigation strategies: The role of access control and privileged access management. In Proceedings of the 2015 IEEE Trustcom, Washington, DC, USA, 20–22 August 2015. Available online: https://ieeexplore.ieee.org/abstract/document/7345393/ (accessed on 3 December 2015).
2. Uber's Massive Hack: What We Know. Available online: https://money.cnn.com/2017/11/22/technology/uber-hack-consequences-cover-up/index.html (accessed on 23 November 2017).
3. Wang, G.; Liu, C.; Pan, H.; Fang, B. Survey on Insider Threats to Cloud Computing. *Chin. J. Comput.* **2017**, *40*, 296–316.
4. Cappelli, D.M.; Moore, A.P.; Trzeciak, R.F. *The Cert Guide to Insider Threats: How to Prevent, Detect, and Respond to Information Technology Crimes*; Addison-Wesley Professional: Boston, MA, USA, 2012; pp. 1–20.
5. Zhang, X.-L.; Wu, X.-Y.; Zhang, W.-X. Research and Implementation of RDP Proxy Proxy-based Audit System. In Proceedings of the CNCT 2016, Xiamen, China, 16–18 December 2016. Available online: https://www.atlantis-press.com/proceedings/cnct-16/25870815 (accessed on 1 January 2017).
6. Cui, W.; Li, H.; Li, W.; An, S. The design and implementation of remote desktop access audit system. In Proceedings of the ICCCT 2012, Seoul, Korea, 3–5 December 2012. Available online: https://ieeexplore.ieee.org/abstract/document/6530527/ (accessed on 13 June 2013).
7. Dusi, M.; Napolitano, S.; Niccolini, S.; Longo, S. A closer look at thin-client connections: Statistical application identification for qoe detection. *IEEE Commun. Mag.* **2012**, *50*, 195–202. [CrossRef]
8. Suznjevic, M.; Lea, S.K.; Humar, I. User behavior detection based on statistical traffic analysis for thin client services. *New Perspect. Inf. Syst. Technol.* **2014**, *2*, 247–256.
9. Bhaskaran, K.; Hernandez, M.; Laredo, J.; Luan, L.; Ruan, Y.; Vukovic, M.; Driscoll, P.; Miller, D.; Skinner, A.; Verma, G.; et al. Privileged identity management in enterprise service-hosting environments. In Proceedings of the IEEE/IFIP NOMS 2012, Maui, HI, USA, 16–20 April 2012. Available online: https://ieeexplore.ieee.org/abstract/document/6211991/ (accessed on 7 June 2012).
10. Tan, Z.; Wu, X.; Wen, Q.; Zhang, H. Design and implementation of proxy- based SSO and security audit system for remote desktop access". In Proceedings of the AIAI 2010, Beijing, China, 23–25 October 2010. Available online: https://https://digital-library.theiet.org/content/conferences/10.1049/cp.2010.0783 (accessed on 20 January 2011).
11. Regateiro, D.D.; Pereira, Ó.M.; Aguiar, R.L. Server-Side Database Credentials: A Security Enhancing Approach for Database Access. In Proceedings of the DATA 2017, Madrid, Spain, 24–26 July 2017. Available online: https://link.springer.com/chapter/10.1007/978-3-319-94809-6_11 (accessed on 30 June 2018).
12. Regateiro, D.D.; Pereira, Ó.M.; Aguiar, R.L. SPDC: Secure Proxied Database Connectivity. In Proceedings of the DATA 2017, Madrid, Spain, 24–26 July 2017. Available online: https://link.springer.com/chapter/10.1007/978-3-319-94809-6_11 (accessed on 24 July 2017).
13. Sade, Y.; Adar, R. Methods and systems for solving problems with hard- coded credentials. U.S. Patent8 468 594, 18 June 2013.
14. Dinoor, S. Privileged identity management: Securing the enterprise. *Netw. Secur.* **2010**, *12*, 4–6. [CrossRef]
15. Sindiren, E.; Ciylan, B. Privileged Account Management Approach for Preventing Insider Attack. *Int. J. Comput. Sci. Netw. Secur.* **2018**, *18*, 33–42.
16. Walker, P. Why do PAM projects fail? *Netw. Secur.* **2019**, *9*, 15–18. [CrossRef]
17. Khaliq, S.; Tariq, Z.U.A.; Masood, A. Role of User and Entity Behavior Analytics in Detecting Insider Attacks. In Proceedings of the 2020 International Conference on Cyber Warfare and Security (ICCWS), Islamabad, Pakistan, 20-21 October 2020; pp. 1–6. [CrossRef]
18. Qiao, J. Research and Implementation of Proxy and Forwarding of RDP protocol. Master's Thesis, Dept. Computer science and technology, North China Electric Power University, Beijing, China, 2015.
19. Yao, Y. VNC-Based Operation and Maintenance Graphical Event Analysis system. Master's Thesis, Dept. Electronic Science and Technology, Dalian Maritime University, Dalian, China, 2012.
20. Tirtadjaja, W.; Rana, M.E.; Shanmugam, K. Managing High Privileged Accounts in IT Enterprise: Enhanced Security Infrastructure. In Proceedings of the 2021 International Conference on Data Analytics for Business and Industry (ICDABI), Sakheer, Bahrain, 25–26 October 2021; pp. 655–660.
21. Wu, D.; Wang, R.; Dai, P.; Zhang, Y.; Cao, X. Deep strip-based network with cascade learning for scene text localization. In Proceedings of the ICDAR 2017, Kyoto, Japan, 9–15 November 2017. Available online: https://ieeexplore.ieee.org/abstract/document/8270071/ (accessed on 29 January 2018).
22. Tian, Z.; Huang, W.; He, T.; He, P.; Qiao, Y. Detecting text in natural image with connectionist text proposal network. In Proceedings of the ECCV 2016, Amsterdam, The Netherlands, 8–16 October 2016. Available online: https://link.springer.com/chapter/10.1007/978-3-319-46484-8_4 (accessed on 17 September 2016).

23. Shi, B.; Bai, X.; Yao, C. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE PAMI* **2017**, *39*, 2298–2304. [CrossRef] [PubMed]
24. Lee, C.Y.; Osindero, S. Recursive recurrent nets with attention modeling for OCR in the wild. In Proceedings of the IEEE CVPR 2016, Las Vegas, NV, USA, 27–30 June 2016. Available online: https://openaccess.thecvf.com/content_cvpr_2016/html/Lee_Recursive_Recurrent_Nets_CVPR_2016_paper.html (accessed on 12 December 2016).
25. Zhou, Y.; Ye, Q.; Qiu, Q.; Jiao, J. Oriented response networks. In Proceedings of the IEEE CVPR 2017, Honolulu, HI, USA, 21–26 July 2017. Available online: https://www.computer.org/csdl/proceedings-article/2017/cvpr/0457e961/12OmNrJAebr (accessed on 9 November 2017).
26. Graves, A.; Fernández, S.; Gomez, F.J.; Schmidhuber, J. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In Proceedings of the ICML 2006, Pittsburgh, PA, USA, 25–29 June 2006. [Online]. Available online: https://dl.acm.org/doi/abs/10.1145/1143844.1143891 (accessed on 25 June 2006).
27. Shamir, A. How to share a secret. *Commun. ACM* **1979**, *22*, 612–613. [CrossRef]
28. Sebastian, S.A.; Malgaonkar, S.; Shah, P.; Kapoor, M.; Parekhji, T. A study & review on code obfuscation. In Proceedings of the Startup Conclave 2016, Coimbatore, India, 29 February–1 March 2016. Available online: https://ieeexplore.ieee.org/abstract/document/7583913/ (accessed on 6 October 2016).
29. Wolf, C.; Jolion, J.M. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *Int. J. Doc. Anal. Recognit.* **2006**, *8*, 280–296. [CrossRef]
30. Yang, F.; Tschetter, E.; Léauté, X.; Ray, N.; Merlino, G.; Deep, G. Druid: A real-time analytical data store. In Proceedings of the ACM SIGMOD 2014, Snowbird, UT, USA, 22–27 June 2014. Available online: https://dl.acm.org/doi/abs/10.1145/2588555.2595631 (accessed on 18 June 2014).