PM-IUBC: A P2P and MongoDB based Intranet User Behavior Control System

Qinghe Dong[1, 2], Qian He[1,3], Huaxin Chai[1], Yong Wang[1,2], and Shengtao He[1]
[1] Key Lab of Cloud Computing and Complex System, Guilin University of Electronic Technology, Guilin 541004, Guangxi, China
[2] Guangxi Key Lab of Automatic Detecting Technology and Instruments, Guilin University of Electronic Technology, Guilin 541004, Guangxi, China
[3] School of Computer Science, University of Manchester, Manchester, M13 9PL, UK
Email: daphny@guet.edu.cn; treeqian@gmail.com; chxgliet@gmail.com; wang@guet.edu.cn

Abstract—It is important for a government unit or a company to monitor the intranet security and staffs' activities during the working time. Collecting and processing massive user behaviors is difficult in the distributed intranet with thousands of hosts. In view of this, this paper proposes a P2P and MongoDB based Intranet User Behavior Control system (PM-IUBC) which consists of the user terminal, the node sliding server overlay, the database cluster and the monitor center. The user behavior control is implemented on the filter driver embedded in the user terminal, and a slice node server overlay is constructed on Pastry. A uniform light weight communication framework is designed to simplify network programming, and a MongoDB with auto shard based log database is configured to store massive user behavior logs. The user behaviors about operating software applications, removable storage devices and files can be monitored and controlled. Experiment results show that PM-IUBC can control the user behaviors and has good concurrency and scalability.

Index Terms—User behavior control, intranet security, P2P overlay, filter driver, MongoDB

I. Introduction

With the development of information technology, computer network has become a necessary part in people's daily lives, and nearly every government unit or company has its private internal network. Company managers normally care very much about the staffs' working efficiency and the internal network security. Investigations of [1], [2] show that about half of office workers spend a lot of time in their private pleasures, such as shopping online, socializing on Face book, QQ, and e-game, and about 14% of UK workforce spends almost half of their work time on the Internet for personal use. These misbehaviors will decrease the working efficiency. In addition, some confidential information including business plan, customer information, and source code may be leaked from the internal network, and the misbehaviors may cause the whole enterprise local network become more vulnerable by worms, Trojans, and viruses. Thereby, monitoring user behaviors for the internal network becomes more and more important for units and companies.

The intranet user behavior control/monitor application needs to control/monitor the user behaviors in the intranet of a specific organization unit or a company. At present, there are many user behavior monitor applications such as WebWatcher [3], SurfControl Web filter [4], Beijing Topsec Network Guardian Gateway System [5] etc. These applications filtering network stream can prevent from data theft, monitor internet and multiple device, and track the behavior of internet users at your convenience from any internet connected computer and disable their misbehaviors [3], [6]. Due to most traditional user behavior monitoring applications work in the network layer, the user behaviors in the host application layer such as opening software, accessing removable storage devices and files which are tightly based on the Operation System (OS) cannot be monitored. Without the monitoring about the user behaviors in the host application layer brings a great deal of potential risks. If the monitoring mechanism is based on the lower layer of OS, it is better to implement fine-grained control [7].

A modern OS uses layered file system modules to support I/O packet flow, and then software developers can intercept and change operations in the packet flow procedure [7]. The filter driver working in the underlying operation system can monitor any file operation from the application layer through checking the I/O Request Packet (IRP) in the Microsoft Windows Systems, so the monitor based on filter driver can control nearly all the operations about software, removal storage and files. The filter driver technology is widely used in the antivirus utilities, encryption programs, and hierarchical storage

Manuscript received March 8, 2015; revised December 8, 2015.
This work was supported by China Scholarship Found under Grant No. 201208455032, the National Natural Science Foundation of China under Grant No.61201250, 61172053, the Guangxi Natural Science Found under Grant No.2012GXNSFBA053174, the Guangxi University Key Lab of Cloud Computing & Complex System Found under Grant No.14102, the High Level of Innovation Team of Colleges and Universities in Guangxi Outstanding Scholars Program Funding, and the GUET Cloud Security and Cloud Service Innovation Group Project.
Corresponding author email: treeqian@gmail.com
doi:10.12720/jcm.10.12.1027-1036

management systems [8], [9], and it is also feasible in the user behavior control system.

In this paper, a P2P and MongoDB based Intranet User Behavior Control system (PM-IUBC) is proposed. PM-IUBC consists of user terminal, slice node server overlay, database cluster, and monitor center. The user terminal uses filter driver to monitor and control user behaviors including running software, accessing removable storage device, and operating file in real time. The slice node server overlay is constructed with P2P to improve system scalability and load balance capacity. The communication implementation of different network components is based on a uniform light weight communication interface library which simplifies network programming. The data cluster is composed by multiple servers running MySQL and MongoDB databases. MySQL is used to store Meta data and user information of PM-IUBC, whereas, the MongoDB database is mainly used to store behavior logs. The MongoDB with auto shard mode is deployed to support saving and querying massive user behavior logs with high performance. PM-IUBC is developed using C++ and C#, and the user behavior controlling functions are evaluated and the performances are analyzed by experiments.

The rest of this paper is organized as follows. Section II presents the related works. The system architecture of PM-IUBC is given in Section III, Section IV discusses how to realize the user behavior control and Section V proposes the uniform communication framework, pastry based node sliding server communication and a MongoDB with auto shard based log database. The performance is evaluated in Section VI and we conclude the paper in Section VII.

II. Related Work

The user behavior control system for the intranet mainly includes network monitoring, application software control and removable storage device monitoring. Accurate and efficient monitoring is vital to ensure that the network operates according to the intended behavior and then to troubleshoot any deviations. The current practice of user behavior control largely depends on manual operations, and thus enterprises spend a significant portion of their budgets on the workforce that monitor their networks [10]. Reference [10] analyzes present network-monitoring technologies, identify open problems, and suggest future directions. In the network behavior monitoring, some commercial filtering systems at present are developed using either source-based blocking techniques or content-based filtering techniques.

The source code based filtering systems rely on a pre-defined white list and black list, where these lists are needed to be updated frequently. In the content-based filtering systems, the loaded contents are compared with pre-defined keywords, phrases and profiles and then determine this web page can be open or not. To some extent, it can effectively prevent web users from unhealthy web contents, however they may not detect the sensitive data which are encrypted or obfuscated, and cannot deal with some images without text explanations or with hidden text notes [11]. Reference [12] proposes a method by integrating the BHO (Browser Helper Object) plug-in techniques with URL-based filtering techniques, which can not only filter out objectionable websites, but also record users' online behaviors into XML documents. Reference [13] puts forward to using windows driver combined with API function to write the virtual equipment articulated driver. This method can track the enterprise internal file access and make detailed monitoring records. The monitoring based on Windows message mechanism can only monitor the relative file information operated through the resource manager. The file operation run in the other background programs cannot be monitored and intercepted before illegal operation. The file filter driver can solve this problem, which can capture the file operation in the whole system, and determine the interception or not according to rules. A network terminal device information anti-leakage system is proposed to prevent the leakage of confidential information in [13]. For small business networks, Reference [14] compares and discusses some File Integrity Monitoring (FIM) techniques and provides an FIM solution.

It is very important for Internet security in business to control and manage user behaviors and the usage of removable storage device for the USB based removable storage device is a main form to leak commerce secrets. The traditional methods are shielding USB port physically and modifying BIOS or Registry. This method can remove the security hazards brought by USB storage devices, but it can also bring lots of inconvenience to our work or lives. One method is encrypting secret information, which is very inconvenience to user to encrypt every file that need protected and cannot prevent some hack attack by ferry-horse. Another method is encrypting the boot sector to aim at control the access policy of USB storage devices [15]-[17]. The traditional monitoring software of USB storage device is developed based on user-level, which is easily bypass by Trojan horses, viruses and other malicious programs, but cannot complete the process of monitoring effectively and real timely.

The user behavior logs must be recorded in order to effectively to be checked in the future, so massive storage logs with numerous users should be stored. The user behavior log management should has three characteristics: 1) high concurrent reading and writing performance; 2) fast saving and accessing huge amounts of data, and 3) high scalability and availability for massive data. The traditional relational database needs to spend a high amount of processing time to achieve these characteristics. The traditional relational database cannot satisfy with the demands, but NoSQL database provides a simple, lightweight mechanism for storage and retrieval of data that provides higher scalability and availability than traditional relational databases. Nowadays, NoSQL database systems are widely used by such big giant companies such as Google, Amazon, and Face book [18]-[20]. MongoDB is an open source document-oriented NoSQL database system written in the C++ programming language, which supports auto-sharding to eliminate manual sharding. Therefore, MongoDB is a better solution for massive storage log in the user behavior system.

Since computers in the intranet are decentralized and difficult to be monitored, and numerous users may generate massive monitor logging data, current network monitor system cannot control user behaviors well and a novel mechanism is needed.

III. System Architecture and Model

The system architecture of PM-IUBC is distributed, where the user terminal, the node slice server, the database cluster, and the monitor work together to realize a scalable and high performance system for massive intranet user behaviors. The design requirement, the system architecture, the filter driver based monitor model, and the P2P based slice node server overlay are introduced.

A. Design Requirments

The intranet user behavior control system is a typical distributed network program with massive network interactions. There are some features and design requirements as following:

1. The environments of the monitored user and computer are very complicated, and user behaviors information on each computer host is difficult to be collected and controlled reliably. The client monitor program should work in a low layer mode of OS that may not be lingered by users and can control user behavior promptly.

2. Thousands of computers are monitored in an internal network, and the information processing load is high. So a scalable distributed architecture is necessary to balance the system's load and to support increasing working capacity dynamically with the load.

3. Large scaled user behavior logging information is generated every minute, and these logs are not changed after collected. Therefore, a scalable database with inserting speed and Query Per Second (QPS) are important, but the updating speed of logging database is not needed to be paid more attention.

4. In a network program of user behavior control system, it may cost much developing time to use socket application interface given by the default library in network program directly. A uniform communication can simplify system realization and assure each component's performance.

B. System Architecture

Fig. 1 illustrates the system architecture of PM-IUBC which is composed of the user terminal, the node slice server overlay, the database cluster and the monitor center.

The user terminal, installed on the users' computer, is responsible for collecting user logs and doing the command from the monitor center. The behavior hardware and software is controlled based on the administration rules commanded by the monitor center. Considering the Microsoft Windows Operation System (OS) is the most popular used system of common users, the OS about the user terminal is mainly based on Microsoft Windows.

The node slice sever overlay is responsible for data acquisition and control from user-controlled terminal, and then allocating more than one slice node servers with value to client in order to communicate with one slice node corresponding to the hash value, which depends on client information, the load of every slice node and load balance algorithm. The slice node severs constructed on a P2P overlay are to receive the data from client and then provide services for client, that is, each node just process a part of user terminals' request.

The database cluster consists of MySQL and MongoDB servers. The system configuration parameters and user information are stored in MySQL, whereas, the received massive logs are stored in MongoDB.

The monitoring center is a web server application, which provides management and monitor platform for the administrator. Besides the user behavior monitoring, the monitoring center also can monitor the whole running status of PM-IUBC.

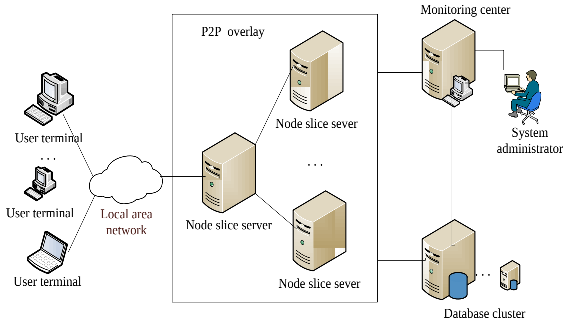


Fig. 1. System architecture of PM-IUBC

C. Filter Driver based Monitor Model

The Microsoft Windows is currently the most widely used operating system for normal users, so the user terminal is discussed mainly for the Microsoft Windows operating system in this paper. The user terminals are implemented based on filter driver which is fast, stable. The filter driver technology can solve some problems existed in Windows API-Hook mode which cannot fully monitor system operation from daemon and bypass problem [21], [22]. The filter driver based monitor model is composed of the file filter driver and the control agent, and the monitor principles are shown in Fig. 2.

In the filter driver modules, Network Driver Interface Specification (NDIS) and file system filter driver are used for appropriate device control object to handle IRP by control rules. The user behaviors such as online activities, running software application, accessing removable storage device and files can be monitored and controlled after the techniques of NDIS and file filter driver. The filter driver module can achieve personalized demand management monitoring by setting time and content pattern. The user network behavior monitoring is based on NDIS like the traditional network firewall. We focus on the user behavior monitor based on file filter driver in this paper.

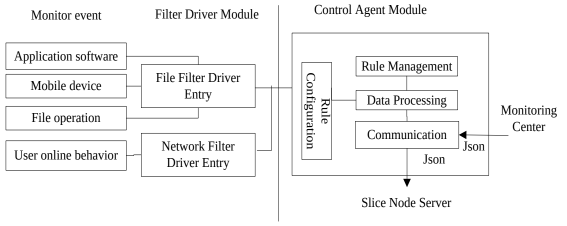


Fig. 2. Filter driver based monitor model

The control agent module consists of four subcomponents: rule management, data processing, EasySocket communication and rule configuration. The commands come from monitoring center, and these commands are transmitted to filter driver module by *DeviceIoControl* after the control rules translation and filtering process. The control rule may prohibit specific URL operations, application programs, or removable devices. The communication messages from the monitor center are described by the Json format. The control results, i.e. the user behavior logs, are returned by the filter driver module, and the control agent submits these user behavior logs to the monitor center through the slice node overlay in PM-IUBC.

D. P2P based Slice Node Server Overlay

The P2P based slice node server overlay is a middle ware to forward and process the messages between the user terminals and the monitoring center or the log databases. Such structure P2P protocols as Chord [23], CAN [24] and Pastry [25] can be used as the routing protocol. In the overlay, each node is similar and can be replaced each other if any slice node is down.

Since Pastry considers the node location information, it is selected to be used in the node slice server overlay in our PM-IUBC. The corresponding slice node for a user terminal is selected by the user terminal's *ID*. Assuming the number of a slice node is *K*, we can get the routing process as follows. Firstly, if the contacted slice node is not responsible for that terminal, the request messages will be forwarded to the node having the nearest distance from node *K* in its neighbor set. If the sending is successful and ACK is received from the closer node, the response message will be sent and tell the terminal *ID* to contact the exact node slice server $K^{'}$. Otherwise, it will be retransmitted again within limited times. If the closer node cannot be found in the neighbor set, then the length *m* of the similar prefix, which both the *ID* of the home node and *K* have, is calculated. If a node with similar prefix in the routing table exits, the routing message is forward to that node. If the node doesn't exist, the message is forwarded to the node which the length of common prefix is *m*, and this node $K^{'}$ is closer to *ID* than the node *K*. The routing process is run in the link establishment for the user terminals. The monitor center knows the corresponding slice node for each terminal, and then the monitoring center can communicate with the correct node slice server directly. The detailed P2P based communication mechanism for slice node server overlay is introduced in Section V.

IV. Realizations of User Behaviour Control

The user behavior control is mainly implemented in the user terminal. The control rule management and the user behavior control about running software, using removable storage device, and file operation are discussed in this section.

A. Control Rule Management

The user terminal receives user behavior command from the monitor center, so the original control rules are defined by the system administrator of PM-IUBC. The control agent module embedded in the user terminal has a control rule management sub-module to cache the rules from the monitor center, and provides local rules adjustment capability. Two role lists: the black-list and white-list are defined by local user or system administrator of PM-IUBC. The definition fields of rules consists of event type, keyword, defined time and match policies including "Equal", "StartsWith", "EndsWith", or "Case Sensitivity". The rule policies are different for different user behavior control.

The rule matching process exists in both the control agent and filter driver module and different user behavior monitor is realized with different matching functions. After capturing the event in the filter driver module of the user terminal, the control rule management component processes the control rules and IRP. Then the result is determined based on the rule matching that if an action is legal, it will pass on to the application layer, otherwise it should be blocked. The control rule management can collect the rule matching result as logs. In the matching process, the white-list is preferred to the black-list rule, i.e. if one action is permitted in the white-list, it can pass directly.

B. Application Software Control

It is important to monitor and control the running software application of computers in the intranet. The system administrators can control when and what program can be run on the computers through the monitor center. When a program is running by a user, the application software control function embedded in the filter driver module can find this running event and the corresponding program, and then do operations based on the rule policies defined in last sub section. The processes of application software control are as follows.

- Registers callback function *m_NotifyRoutine* and calls *PsSetCreateProcessNotifyRoutine* in the driver entry function *DriverEntry*;
- Determines whether or not to create a process in the function *m_NotifyRoutine*. If yes, send the process ID and parent process ID to the control agent module.
- Obtains the name of the new run software by the process ID in the control agent module of the user terminal, and then submits the current time, parent process and child process name to the control rule management to evaluate this running software action is legal or not.
- Determines kill or continue the new run program in the control rule management. If it is not legal, the program is killed, otherwise, it is continued. Finally, the log information is submit to to the P2P based slice server overlay.

C. File Operation Control

The file operation control tries to monitor and control the user behaviors about operating files in the intranet. When a file or folder is opened, read or written by an application, the corresponding IRP is generated by OS which can be captured by the filter driver module. Besides of capturing, the file system filter driver filters IRP, i.e., IRP is passed in the rule matching to be discard or not. So we have the ability to capture and change the file system operations. The entry function *DriverEntry* registers five dispatch functions: *IRP_MJ_CREATE*, *IRP_MJ_WRITE*, *IRP_MJ_READ*, *IRP_MJ_CLOSE and IRP_MJ_SET_INFORMATION*.

Fig. 3 illustrates the process flow chat of *IRP_MJ_CREATE*. In *IRP_MJ_CREATE*, such file operation events as opening, creating, reading, and writing can be captured and filtered based on the control rules. The white_list and black_list are separated based on reading and writing operations. If one of these names is in the white list, then the renaming operation is allowed. Otherwise, reject the renaming operation if one of these file name is in the black list. After matching in the white_list and black_list, the corresponding file operation will be determined to be permitted or rejected.

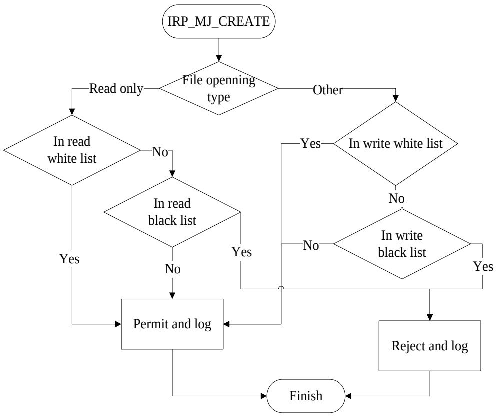


Fig. 3. Dispatch function of *RP_MJ_CREATE*.

Fig. 4 illustrates the process flow chat of *IRP_MJ_WRITE*. The processing procedures are similar to *IRP_MJ_CREATE* except that *IRP_MJ_WRITE* can check a file' writing permission deeply to determine whether the file operation can be permitted or not.

The other functions of *IRP_MJ_READ*, *IRP_MJ_CLOSE and IRP_MJ_SET_INFORMATION* are similar to the above *IRP_MJ_CREATE* and *IRP_MJ_WRITE*. When reading a file, the reading permission of control rules are checked by using *IRP_MJ_READ* function. While deleting a file, modifying the name, the dispatch function *IRP_MJ_SET_INFORMATION* will be triggered. When a user renames a file, the old file name and the new file name are compared.

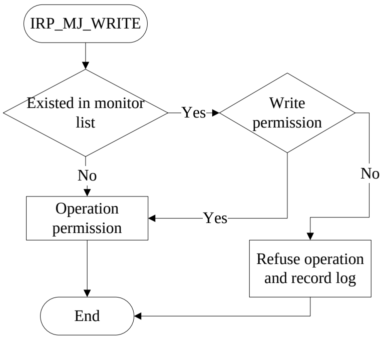


Fig. 4. Dispatch function of *RP_MJ_WRITE.*

D. Removable Storage Device Control

A removable storage device can be set to disable or access control with control rules through the monitor center of PM-IUBC. If a user is not given the permission for a device in the control rules, the removable storage device cannot be used by that user. The removable storage device control helps to restrict user access to removable storage devices illegally or leak the sensitive data of a company. The removable storage device control scan and get the removable storage devices label automatically, and the access control for files in the removable storage can be done depending on the file operation control after generating temporal matching rules for the files in the removable storage. The implementing processes of removable storage device control are as follows.

- Binds the dispatch function *DispatchStart* to *IRP_MN_START_DEVICE* in *DriverEntry*.
- Gets the type of device using *GetDeviceTypeToUse* in *DispatchStart*. If it is a removable storage device, the device information is send to the control module.
- Detects a newly added mobile storage device in the filter driver module of the user terminal. Based on the control rules for the removable storage device, corresponding temporal matching rules are generated automatically. The temporal matching rule is similar to the rules defined by system administrator.
- The next implementation operations are similar to the file operation control discussed earlier.

The removable storage device control can also refer to the article [26] where it was achieved through operating *IRP_MJ_SCI*. However, the file operation monitor is already implemented in the file operation control earlier, so it is natural and easy for the removable storage devices control to implement with the file operation. The novel method has the better fine-grain control than the method in reference [26].

V. Communication and Storage

The communication and storage mechanisms of the user behavior system are the key technologies to realize high performance and scalability. A uniform encapsulated socket interface library is designed to simplify and unify the component realization. The Pastry P2P protocol is used to enhance cooperation of node sliding servers, and the MongoDB with auto shard improves logging performance.

A. Uniform Encapsulated Socket Interface

It may cost much time to develop a network program with the original socket. Considering various components in PM-IUBC need network programming, we design a uniform encapsulated socket interface library for communication between the user terminal, the node sliding server, and the monitor center. Since C# is the programming language used in PM-IUBC, the uniform encapsulated socket interface library is realized for C#. Fig. 5 illustrates the uniform encapsulated socket interface framework.

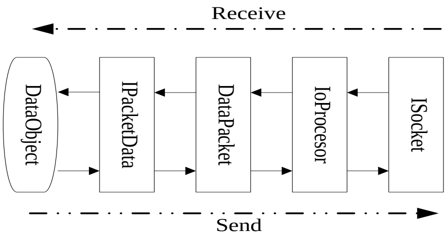


Fig. 5. Uniform encapsulated socket interface framework.

The *IpacketData* class is an abstract class in the message body, inherited from *BsonSerializerBase*. Two main attributes are defined: *HardwareID* and *Type*, where, *HardwareId* represents a unique hardware identification code for a invoking program, and *Type* is the definition of message type.

The *DataPacket* class is a network communication message with object-oriented. Through *DataPacket* class it is easy to translate the received byte stream into an object, and to change object into byte in the network communication.

The *IoProcessor* class is responsible for deciding whether there exist data in the channel need to read or write. While sending data, *DataPacket* is sent to the instance of *IoProcessor*, and then the instance of *IoProcessor* process the data with several modes: synchronous, asynchronous, and communication with reply waiting.

TABLE I: CLASS DEFINITION OF ISOCKET

Isocket Interface
Attributes
CanRead CanWrite Client IsClientMode IsConnected LocalPort RemoteAddress
Functions
Send SendAsync
Events
Received

The *Isocket* class is the main class provided for developers to use to send or receive message in the network program. In *Isocket* class, the function *IoProcessor* is always invoked automatically, and the events of receiving data can be operated in other additional processes. Table I shows the main attributes and functions of *Isocket* class.

B. Pastry Communication for Node Sliding Servers

Multiple node sliding servers are implemented as a P2P overlay with Pastry, which is a typical P2P protocol with local searching ability in the routing. The communication messages exchanging between node sliding servers consist of the monitor center command, the terminal logging and the state information, which have the similar routing procedure. Once a node sliding server receives a communication message, a suitable node is selected comparing the ID of the source or destination host with the ID of the sliding servers. The Pseudo-code of the Pastry communication for node sliding servers is given in Fig. 6.

```
$R_l^l$： node sliding server node with the l throw and the i th
column （$0 \le l < 128/b$，$0 \le i < 2^b$）
$L_i$： The ith server,
uID： ID of the monitored user terminal
1) Receiving message from/to uID；
2)if $L_{-|L|/2} \le uID \le L_{|L|/2}$ then
3)     { Forward $L_i$ ,( $L_i \in L \cap \min(|L_i - uID|)$ )；
4)      if ACK==true then Goto (1)；
5)       else Goto (2)； }
6) else
7)  { $l$ =prefix(uID, $L_i$)；
8)  if $R_l^{uID} \neq NULL$ then {
9)     Forward Sliding Server with ID= $R_l^{K_l}$ ；
10)   if ACK==true then Goto (1)；
11)   else Goto (2)； }
12) else {
13)Forward Server with ID $\in R \cap \min(|R_l^i - K| < |nodeId - K|)$ ；
14)   if ACK==true then Goto (1)；
15)    else Goto (2)； } }
```

Fig. 6. Pseudo-code of the Pastry communication for sliding servers

The pastry overlay provides a communication mechanism for the user terminal to select a node sliding server to upload or download data based on its *ID*. The Pastry communication mechanism is distributed and balanced for the command delivery and data logging collection in PM-IUBC, and the node sliding servers can be added or reduced dynamically on the load of PM-IUBC.

C. MongoDB with Auto-Shard based Log Database

MongoDB is an open source and document based database which is designed for cloud computing and has high performance. Unlike the traditional relational database, MongoDB just stores key value in the collection, so it can keep high response speed for large-scale data management [20], [27].

The MongoDB with auto shard can distribute data over multiple MongoDB nodes. By splitting data over multiple machines, it becomes possible to store more data and handle more load than a single database server [20]. The MongoDB with auto shard based log database provides an excellent platform for statistical analysis, fast query, and disaster recovery. The MangoDB with auto-shard based database for user behaviors log is given in the Fig. 7.

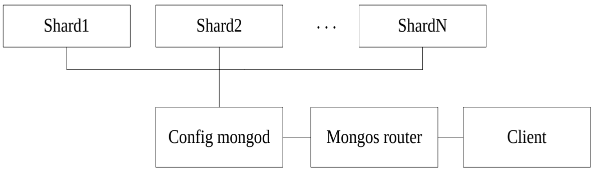


Fig. 7. MongoDB with auto-shard based log database.

Because the user control system in a intranet with hundreds hosts may generate massive log data, PM-IUBC utilizes MongoDB to store user behavior logs,the collection of which includes LOGID, IP, MAC, user name, event type, message content, and occurrence time. The operation on MongoDB database can be achieved through invoking the classes of *MongoDB.Bson* and *MongoDB.Driver* in C# program, and the detailed can be referred to [20]. Based on MongoDB, it is easier to analyze the massive log information or data mining in the later.

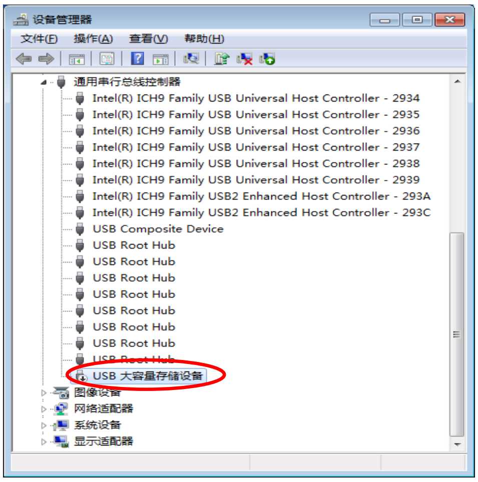
VI. Evaluation

In PM-IUBC, the filter driver module of user terminal is implemented with C language and all the other components are implemented with C#. We deploy PM-IUBC on an internal network with over 200 user terminals, 3 node slice servers, a monitoring center, and 2 database servers. OSes of the user terminals consist of Windows XP, 7 and 8, CentOS 6.3 is installed for the slice and database servers, and the databases are MySQL 5.5 and MongoDB 2.2. The evaluations, including system functions, the slice node server overlay performance and the log database performance, are illustrated.

A. Function Evaluation

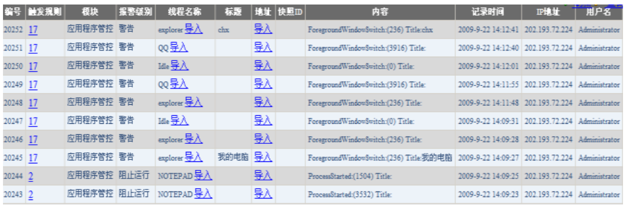
The monitoring center implemented as an ASP.net web server program can manage and control the intranet user behaviors including running software applications and operations files and remote storage devices. The unpermitted behaviors are forbidden and recorded.

According to application software monitor, the system can accomplish software monitor of the different level and different softwares through setting the software process features configure on monitor center. The disabled software is not allowed to open in different prohibited time by the rule management. For example, QQ and popular games can be prohibited during working hours, and then every internal network users can not chat with QQ. The operation information for monitored file will be recorded and the files in read-only folder has limited operation ranks, that is, if a new folder is created, the error message “Destination folder access denied” will be displayed; if delete one file, the error message “File Access Denied” will be shown; if modify the file, the message “Access is denied on xxx” will be prompted. When an unpermitted removable storage device is inserted to the client, it is rejected. Fig. 8 shows that the unpermitted removable storage device in the operation system device management tool is orange and tagged by “!”.



通用串行总线控制器
Intel(R) ICH9 Family USB Universal Host Controller - 2934
Intel(R) ICH9 Family USB Universal Host Controller - 2935
Intel(R) ICH9 Family USB Universal Host Controller - 2936
Intel(R) ICH9 Family USB Universal Host Controller - 2937
Intel(R) ICH9 Family USB Universal Host Controller - 2938
Intel(R) ICH9 Family USB Universal Host Controller - 2939
Intel(R) ICH9 Family USB2 Enhanced Host Controller - 293A
Intel(R) ICH9 Family USB2 Enhanced Host Controller - 293C
USB Composite Device
USB Root Hub
USB Root Hub
USB Root Hub
USB Root Hub
USB Root Hub
USB Root Hub
USB Root Hub
USB Root Hub
USB 大容量存储设备
图像设备
网络适配器
系统设备
显示适配器

Fig. 8. Removable storage device control example



编号	触发规则	模块	报警级别	线程名称	标题	地址	快照ID	内容	记录时间	IP地址	用户名
20252	17	应用程序管控	警告	explorer 导入	chx	导入		ForegroundWindowSwitch:(236) Title:chx	2009-9-22 14:12:41	202.193.72.224	Administrator
20251	17	应用程序管控	警告	QQ 导入		导入		ForegroundWindowSwitch:(3916) Title:	2009-9-22 14:12:40	202.193.72.224	Administrator
20250	17	应用程序管控	警告	Idle 导入		导入		ForegroundWindowSwitch:(0) Title:	2009-9-22 14:12:01	202.193.72.224	Administrator
20249	17	应用程序管控	警告	QQ 导入		导入		ForegroundWindowSwitch:(3916) Title:	2009-9-22 14:11:55	202.193.72.224	Administrator
20248	17	应用程序管控	警告	explorer 导入		导入		ForegroundWindowSwitch:(236) Title:	2009-9-22 14:11:48	202.193.72.224	Administrator
20247	17	应用程序管控	警告	Idle 导入		导入		ForegroundWindowSwitch:(0) Title:	2009-9-22 14:09:31	202.193.72.224	Administrator
20246	17	应用程序管控	警告	explorer 导入		导入		ForegroundWindowSwitch:(236) Title:	2009-9-22 14:09:28	202.193.72.224	Administrator
20245	17	应用程序管控	警告	explorer 导入	我的电脑	导入		ForegroundWindowSwitch:(236) Title:我的电脑	2009-9-22 14:09:27	202.193.72.224	Administrator
20244	2	应用程序管控	阻止运行	NOTEPAD 导入		导入		ProcessStarted:(1504) Title:	2009-9-22 14:09:25	202.193.72.224	Administrator
20243	2	应用程序管控	阻止运行	NOTEPAD 导入		导入		ProcessStarted:(3532) Title:	2009-9-22 14:09:23	202.193.72.224	Administrator

Fig. 9. User behavior records of the monitor center

All the monitored user behaviors are saved to the logging database and can be monitored in the monitor center. A user behavior warning monitor center is given in Fig. 9, where the detailed warning information including category, level, application name, title, content, record time, IP address, and username are recorded.

B. *Slice Node Server Overlay Performance*

In the slice node server overlay, multiple slice node servers work together on P2P overlay which can forward the messages between the user terminal and the monitoring center or the log database. The slice node server overlay work together to process the massive user behavior data. There are 15000 user terminals are simulated in the slice node server overlay performance experiments. Obviously, the 15000 user terminals generate large scalable user behavior log that cannot be processed efficiently by a single slice node. We simulated six slice node servers in the slice node server overlay, and the ID of the slice node server and the user terminal is based on the hash function *SHA-1* for the IP and Port, i.e., *ID* = *SHA1* (*IP+Port*).

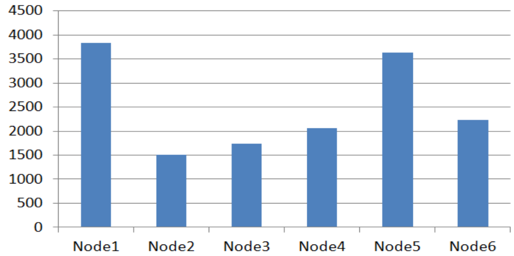


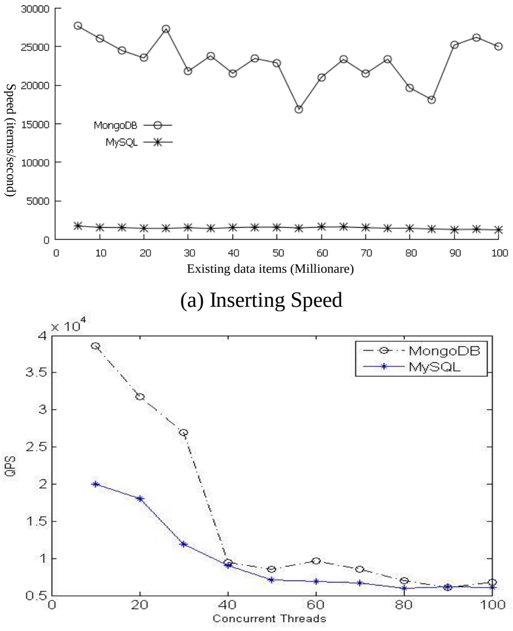
Fig. 10. Loads of slice node server overlay with 6 nodes and 15000 user terminals

Fig. 10 shows the loads of six slide servers in the overlay for 15000 user terminals. the task loads of 15000 user terminals are divided by six slice node servers balancely. If there are more log loads of user terminals, we can add more slice node servers into the overlay to support the increased loads. Since ID of a slice node server is generated by *SHA-1*(IP) when it joined the P2P overlay, the managed ID space of each node is not same, and then the load of each slice node server is not evenly distributed in Fig. 10. Node1 has the highest load with 25.5%, while Node2 just has the lowest load with 10.1%. The ID generating algorithm of the P2P overlay is related with the load distribution a lot, but the load of each node is not equal for a new slice node servers may can break the balance station. Normaly, it is a good method for P2P overlay to arrange a real node to works as multiple virtual nodes as [19].

C. *Log Database Performance*

The log database performance is important for monitoring massive user terminals in PM-IUBC. We deploy the log database on a Dawning A620r-G server (2.1Ghz 32 Cores CPU, 32GB DDR3), and compare the MongoDB log database with the original MySQL. Two metrix: insertting speed and the Query Per Second (QPS) are disscussed. Fig. 11 compares the user behavior log database performance.

At first, the user terminals insert 100 million log items into the log database using 32 threads. The inserting speed is shown in Fig. 11.a. The inserting speed of the original MySQL is stable and the average inserted speed is about 1500 per second. The inserting speed of MongoDB with auto shard is not stable as MySQL, but it can support over average 20,000 per second. The inserting speed of MongoDB log database improved nearly 15 times comparing the MySQL log database. Second, we test the QPS under two log databases with 100 million items and various concurrent threads from 10 to 100. Fig. 11.b shows the QPS results of two log databases. The QPS of the MongoDB is about twice the MySQL's QPS when the concurrent threads is smaller than 30. With the concurrent threads increasing, QPSes of two methods are all decreased, and MongoDB and MySQL have nearly similar QPS when the threads reach 80. We note that the query performance is mainly limited by the input/output of the disk hardware. If we improve the disk processing mode, e.g. using better disk or adding and saving data in different disks, the QPS may improve greatly.



(a) Inserting Speed

(b) QPS under various concurrent threads

Fig. 11. User behavior log database performance comparison

Since MongoDB is designed for distributed computing and has auto shard mode, it is easier to add a new shard server to solve the bottle capacity problem than the traditional database. The whole scalability of PM-IUBC improved greatly for MongoDB log database is used, and PM-IUBC can be suitable to process massive user behaviors' log.

VII. Conclusion

It is important for a government unit or a company to control what users do in the working time and to know the intranet security status, but the computers in the intranet are decentralized and difficult to be monitored. PM-IUBC, a P2P and MongoDB based user behavior control system, is proposed in this paper. PM-IUBC consists of the user terminal, the node slice server overlay, the database cluster, and the monitor center. The filter driver based user terminal can control the user behaviors promptly, the uniform encapsulated socket communication interface simplifies the network programming for PM-IUBC, and the P2P overlay and shard MongoDB technologies help to process massive user terminals and log data fast. The experiments evaluate the main user behavior control functions and the performances of the Pastry based slice node server overlay and MongoDB based log database. The slice node server overlay and the MongoDB log database have excellent scalability to control large scaled user behaviors of the intranet. The user behavior controllability and security in an intranet can be improved after deploying PM-IUBC.

References

[1] Do you surf in work time? [Online]. Available: http://news.bbc.co.uk/1/hi/talking_point/1285181.stm

[2] S. M. Heathfield. What employers are doing about employees surfing the Web at work? [Online] Available: http://humanresources.about.com/od/technology/a/surfing_web.htm/

[3] What is Web Watcher? [Online]. Available: https://www.webwatcher.com/#howitworks

[4] Web Sense Homepage. [Online]. Available: http://www.websense.com

[5] Topsec Homepage. [Online]. Available: http://www.topsec.com.cn/

[6] M. S. Dichter and M. S. Burkhardt. Electronic interaction in the workplace: Monitoring, retrieving and storing employee communications in the internet age. [Online]. Available: http://www.longwoods.com/content/16966

[7] E. Zadok, R. Irer, N. Joukov, *et al.*, "On incremental file system development," *ACM Transaction on Storage*, vol. 2, pp. 1-33, Feb. 2006.

[8] S. Li, Z. Shao, S. Lv, and X. Jia, "The design and implementation of removable storage device monitoring system based on WDM filter driver," in *Proc. International Symposium on Intelligence Information Processing and Trusted Computing*, 2010, pp. 450-453.

[9] Z. Zhao and H. Yao, "A data backup method based on file system filter driver," in *Proc. Second World Congress on Software Engineering*, 2010, vol. 2, pp. 283-286.

[10] S. Lee, K. Levanti, and H. S. Kim, "Network monitoring: Present and future," *Computer Networks*, vol. 65, pp. 84-98, 2014.

[11] X. Zhang, N. Zeng, J. Tang, and X. Xu, "Web user behavior monitoring for campus networks," presented at the IEEE International Conference, 2010, pp. 3828-3833.

[12] X, Tang, Y. Lu, and N. Liu, "Design and implementation for file monitor system based on windows driver," in *Proc. Sixth International Symposium on Parallel Architectures, Algorithms and Programming*, July 2014, pp. 289-292.

[13] J. Qu, Q. B. Li, Y. Bai, *et al*., "Application of file system filter driver in network secure terminal," *Jisuanji Yingyong Journal of Computer Applications*, vol. 27, no. 3, pp. 624-626, 2007.

[14] B. Wilbert and L. Chen, "Comparison of file integrity monitoring (fim) techniques for small business networks," presented at the IEEE International Conference on Computing, Communication and Networking Technologies, 2014, pp. 1-7.

[15] L. Zheng, Z. Ma, and M. Gu, "Techniques of file system filter driver-based and security-enhanced encryption system," *Journal of Chinese Computer Systems*, vol. 28, no. 7, pp. 1181, July 2007.

[16] Z. Gu and H. Huang, "Research and implementation of new encrypting file system," *Computer Engineering and Design*, vol. 30, no. 14, pp. 3272-3277, 2009.

[17] Y. Lin, S. Lin, Y. Li, and L. Shi, "Design and implementation of encryption filter driver for USB storage devices," presented at the computational Intelligence and Design International Symposium, 2011, pp. 356-359.

[18] W. Zhu, M. Li, and H. Chen, "Using MongoDB to implement textbook management system instead of MySQL," in *Proc. IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*, 2011, pp. 303-305.

[19] C. W. Huang, W. H. Hu, C. C. Shih, *et al.* "The improvement of auto-scaling mechanism for distributed database-A case study for MongoDB," presented at the 15th Asia-Pacific Network Operations and Management Symposium on IEEE, 2013, pp. 1-3.

[20] Y. Liu, Y. Wang, and Y. Jin, "Research on the improvement of MongoDB Auto-Sharding in cloud environment" in *Proc. 7th International Conference on Computer Science & Education*, 2012, pp. 51-854.

[21] Y. Wang, Q. He, and S. He, "Research on webpage anti-tampler system base on file filter driver," *Journal of Guilin University of Electronic Technology*, vol. 30, no. 5, pp. 432-435, 2010.

[22] F. Zhang and C. Shi, *Windows Driver Development Internals*, Beijing: Electronic Industry Press, July 2008.

[23] I. Stoica, R. Morris, D. Liben-Nowell, *et al.*, "Chord: A scalable peer-to-peer lookup protocol for Internet applications," *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, pp. 17-32, 2003.

[24] S. Ratnasamy, P. Francis, M. Handley, *et al.*, "A scalable content-addressable network," ICSI Technical Report, San Diego, CA, United states, 2001, pp. 161-172.

[25] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," in *Proc. 18th IFIP/ACM Conference on Distributed Systems Platforms*, Heidelberg (D), 2001, pp. 86-94.

[26] C. Cao, D. Fu, and F. Cao, "Solution of removable storage control based on file system filter driver," *Journal of Computer Applications*, vol. 31, no. 6, pp. 1498-1501, 2011.

[27] P. Murugesan and I.Ray "Audit Log Management in MongoDB" in *Proc. IEEE World Congress on Services*, 2014, pp. 53-57.

Qinghe Dong was born in April 1978, in Biyang County, Henan Province, China. She received BS and MS degree in Guilin University of Electronic Technology, Guilin, China, in 2001 and 2006 respectively. She is currently a lecture at Key Laboratory of Cloud Computing and Complex System, Guilin University of Electronic Technology whose research interests lie in network intelligence and wireless sensor network.

Qian He is a Professor at Guilin University of Electronic Technology, China, and a Visiting Research Associate in the School of Computer Science, University of Manchester, UK. He received B.Sc. from Hunan University, Changsha, China, in 2001, M.S. in Guilin University of Electronic Technology, Guilin, China, in 2004, and Ph.D. from Beijing University of Posts and Telecommunications, Beijing, China, in 2011, and finished the post doctor research in the National University of Defense Technology, Changsha, China. His research interests include network security and distribute computing. He is a member of the ACM, IEEE and CCF.

Huaxin Chai received his MS in Guilin University of Electronic Technology, Guilin, China, in 2006. He is currently an Engineer at Guilin University of Electronic Technology. His research interests lie in network management.

Yong Wang received his PhD in East China University of Science and Technology, Shanghai, China, in 2005. He is currently a Professor at Guilin University of Electronic Technology, and serves as head of the Guangxi Engineering Technology Research Center of Cloud Security and Cloud Service. His research interests include network security and cloud computing.

Shengtao He received his MS in Guilin University of Electronic Technology, Guilin, China, in 2013. He is currently an Engineer at Guilin University of Electronic Technology. His research interests lie in information system management.