An Internet Behavior Management System based on Android

Miao Liu
School of Computer Science and Educational Software/Guangzhou University, Guangzhou, China
Email: liumiao@gzhu.edu.cn

Gengtong Hou, Ying Gao, Chunming Tang, Dongqing Xie
School of Computer Science and Educational Software/Guangzhou University, Guangzhou, China
Email: {hougengtong,gaoying,tangchunming}@ gzhu.edu.cn

***Abstract*— With the popularity of smart phones and mobile Internet, how to prevent young people from being addicted to Internet and games has become one of the issues which are most concern to parents. To solve this problem, we design and implement an Internet behavior management system which is based on Android. The system implements the mobile phone remote control, network interception and application running control. By using the system, users can filter communication data which contain sensitive words out of children's mobile phones, monitor behavior of software in the children's mobile phones, and limit the running time of them.**

***Index Terms*—Android, Internet management, network interception**

I. Introduction

Most current desktop applications of Internet behavior management are designed to limit the online behavior of employees in order to protect corporation's information security. The function of this kind software includes the prohibition or monitoring something such as BT, stocks, chat, MSN, QQ, monitoring e-mail and bandwidth, in order to reduce viruses infecting and achieve the purpose of the correct guidance of employees Internet usage[1-5]. However, mobile devices have small memory and limited resources, what's more, for security reason, most of mobile operating systems have strict restrictions on the functionality of applications and permission management, so it is difficult to obtain permission to manage any application such as stocks, QQ. These make the behavior management software extensions on mobile devices not be implemented so easily like desktop applications, that is the reason why techniques in desktop Internet behavior management software are not entirely suitable for use in mobile devices [6-9].

The rest of the paper is organized as follows: Section II discusses the related work and show the novelty. Section III analyses requirements. Section IV describes the system architecture. Section V puts forward the communication protocol. Section VI discusses the message dispatching mechanism. Section VII describes the system implantation in detail. Section VIII concludes the paper.

II. Related work

There are some mobile parent control applications which are similar to Internet behavior management software, such as "Net nanny", "Norton Family parental control" and etc. This kind of software can block or filter web content, monitor SMS and call, remote control, real-time GPS recording. Their functions are just refined or reduced functions of desktop's parent control software, not bind them tightly with underlying mobile OS. Most parent control applications are just one application installed on the controlled device. Some applications need to set manually on the installed device. Others have remote administration through web sites, and parents also can read reports through these sites.

Based on studying many desktop monitoring systems, we studied deeply on the phone mobile network technology, network interception technology, communication and control, mobile embedded system operating mechanism and other aspects of mobility, designed and implemented an Internet behavior management system. The system is also optimized for Android mobile phone platform [10-20]. The innovative points are listed below.

- **Remote Control**. Unlike the above parent control applications, the system includes the control side application and the controlled side application. One control side may control many controlled sides. All of them are installed in mobile phones. So, parents can control kid's phones through their phones conveniently in real time manner. Parents can set kid's phones alarm events to remind their children, set blacklist/whitelist and also get usage or alarm reports in real time.
- **Intercept deeply**. The current parent control applications can just block and filter web contents. We develop our intercept module combining with Linux kernels, and can intercept any communication data into or from any applications on the control sides.

doi:10.4304/jsw.8.8.1932-1942

- **Applications control.** Due to security guarantee, the top API of Android OS does not provide any functions about applications control. We use Linux kernels to implement opening and closing any applications on the controlled sides. So, parents can set starting time, time intervals and span limits for applications of kid's phones.

III. Requirements Analysis

For the needs of behavior monitoring on phones of kids, this software's requirements are listed below.

A. Functional requirements

- The control side should be able to instantly communicate with the controlled side.
- The control side should be able to monitor and intercept the text messages on phones of the controlled side, and can edit interception list on phones of the controlled side. The controlled side doesn't have permission to modify this list.
- The control side should be able to add alarm event to alert important events to the controlled side.
- The control side should be able to monitor and control the running time of all applications on the controlled side, and the controlling rules can be set by the control side.
- The control side should be able to monitor and filter network information on the controlled side, and the filtering rules can be set by the control side.
- The control side should be able to monitor the GPS position of the controlled side, and the control side should be able to display this location information.
- When phones of the controlled side has been replaced the SIM card, the control side should be notified.
- When the controlled side program is uninstalled, clear the data or forced to shut down, the control side should be notified.

B. Non-functional Requirements

- From the software's easy-to-use of consideration, the user interface design should be simple and features should be obvious in the GUI.
- To consider the reliability of the software, the functionality of the program should be able to run properly when not connecting to the server.

IV The System Architecture

The system is divided into four components, the control terminal program (hereinafter referred to as the control side) is running on phones of parents, the main program of the controlled side (hereinafter referred to as the controlled side) and protection procedures (auxiliary controlled side) are running on phones of kids, server-side program (hereinafter referred to as server side) is running on a public network server. The control side is responsible for sending control instructions to the controlled side. These instructions include setting the alarm, application running time control, location monitor, and network interception and so on. The main program of the controlled side is responsible for the execution of the received instructions and feeding the results back to the control side. These results include the location information, the content of the SMS which has been intercepted, and so on. The saver on the controlled side is responsible for monitoring the operational status of the main program. If the main program is forced to stop or uninstall by users, the saver will promptly notify the control side, in order to prevent the behavior of closing the program of controlled side by the children to escape from monitoring of their parents. The server side is responsible for forwarding the communication information between the control side and the controlled side, in order to achieve cross-Net communication.

The programs on the Android platform (the control side and the controlled side) have the similar structure of the code. The code packet divides into four parts: user interface, model, utility and bean. The user interface part includes the interface design. The model part includes the implementation of the main logic functions, such as the alarm function and network interception function. The utility part includes the implementation of some common functions and interfaces which can be used by other parts. The bean part includes definitions of some simple data structures which will be used in communication processes. The packet diagram is shown in Figure 1.

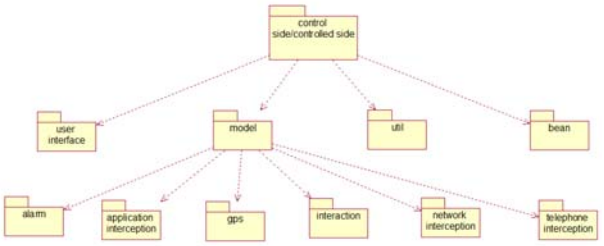

Figure 1 The system packet diagram.

Each component (the control side, the controlled side) in the system has adopted a hierarchical system structure. From top to bottom are the view layer, Android component layer, model layer, tool layer and kernel layer. Classes in the View layer are mainly responsible for the user interface and responding to user action. Classes in the Android component layer are responsible for the interaction with system services, such as listening SMS. Classes in the Model layer are mainly responsible for the handling of the users' data, such as alarm clock event management. Classes in tool layer provide some general function such as command interface operation and format conversion. The Kernel layer runs a module-NetHookModule, which monitors the flow of data from the network in the system kernel. System layered architecture diagram is showed in Figure 2.

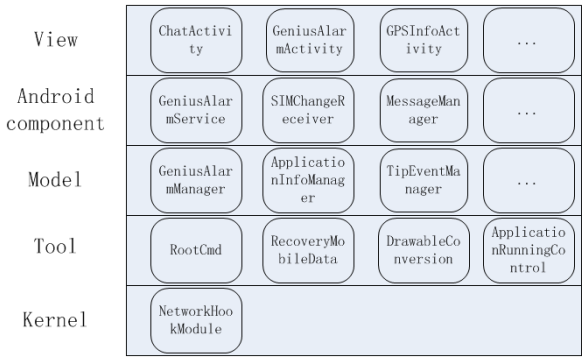

Figure 2 System layered architecture.

When we start the application on Android phones, we first see a graphical UI interface. When we touch its screen or press button, the components in the UI will be called the initialization operation and functional operation of the respective module, then jump into the respective interface of each module. In the controlled side, the main program maintains the Android underlying kernel synchronization and feedback, and also maintains the synchronization with the controlled side saver.

The interaction among components in the system is shown in Figure 3.

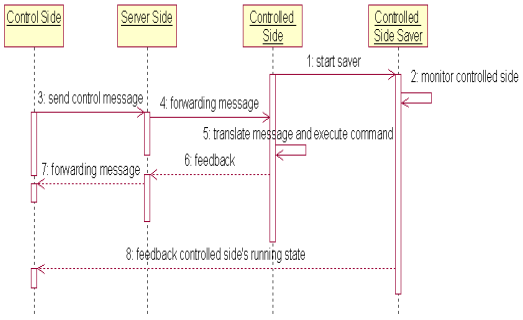

Figure 3 System sequence diagram.

The controlled side automatically starts the saver at boot time. The saver is responsible for monitoring running states of the controlled side programs. If the controlled side programs close unexpectedly or are uninstalled, the saver will notify the control side promptly to ensure the control side to deal with the situation. The control side sends control message to the controlled side through the server side, such as accessing the application information on controlled side, and so on. The controlled side receives the control message, the execution results will feedback to the control side by communicating with the server side, and so the controlled side can communicate with the control side only if each keeps the connection with the server side. But when the main program of controlled side is abnormal, the saver can communicate with the control side directly, so the abnormal state can be notified to the control side promptly.

The system adopts the C/S architecture, the control side and the controlled side is client sides, and the server side is the server component. Two clients keep their communication through their connections to the server.

V. THE COMMUNICATION PROTOCOL

It is very important for message recognition and execution during communication processes, so it must have an agreement to keep the information in the identification process on each side. The protocol ensures that the receiver side can smoothly perform the intent of the sender side. It is necessary to ensure the consistency of the data definition. In the system, there is a common package, which defines the communication data format. The communication protocol is divided into four parts: the packet format, request codes, response codes and the message format.

A. The Packet Format

Communication packet format is shown in Table I.

The beginning of data packet is the protocol data unit type, which is identified by request codes or response codes. The following parts are the destination and source addresses, the source and destination addresses are numbers of mobile phones, and the end of a packet is message part. The message is determined according to the first protocol type. When sending a message, the system will fill in the protocol type according to the message type, and when receiving information, the program will parse out the correct message which corresponds to the protocol type. In the system, data packets are transported through TCP and UDP protocols, so the protocol eliminates the need for the data checksum and retransmission mechanism.

TABLE I.
COMMUNICATION PACKET FORMAT

request code/ response code	source phone number	destination phone number	message

B. Request Codes

The command side requests the receiving side to give something of response or execute some function. The request code represents the type of commands. The control side and the controlled side communicate with the server side through the same protocol, this can ensure that both sides of the data transmission and parsing operations are symmetrical, in order to achieve the accuracy of the communication. Table II lists all request codes and their meaning.

C. Response Codes

Response codes are the identification of the feedback of the corresponding command message. When parsing a returning data packet, by extracting the response codes firstly, the system can know the type of packets and submit the data packet to the corresponding message module to interpret its meaning, or prompt the user to some appropriate information, such as interrupted the connection with server and so on. Table III lists response codes and their meanings.

D. Message Format

The message is the last part of data packets. This part needs to be analyzed by the message parsing module. The type of message corresponds to request code / response code. During parsing process, the analysis module can parse out the correct message by the request code / response code. All the request code and the corresponding code is listed in TABLE II and TABLE III, and Table IV lists the message type which corresponding to the requested code / response code, the request codes / response codes not listed in Table IV correspond to the empty message which the parsing module can ignore.

TABLE II.
RESPONSE CODES AND MEANINGS

Response code	Meaning
RESPONE_GPS_INFO_MESSAGE(0x6)	Feedback location information
RESPONE_APP_INFO_MESSAGE(0x8)	Feedback application information
RESPONE_UDP_INFO_MESSAGE(0x9)	Feedback UDP socket information
AGREE_TO_CONNECT_MESSAGE(0xb)	Feedback the result of agreement to connect
DISAGREE_TO_CONNECT_MESSAGE(0xc)	Feedback the result of disagreement to connect
RESPONE_APP_NUM_MESSAGE(0xe)	Feedback the number of the application
GET_GPS_INFO_FAIL_MESSAGE(0xf)	Feedback the result indicates the failure to get the GPS information
RESPONE_APP_ICON_MESSAGE(0x11)	Feedback the icons of the application
RESPONE_SERVER_INTERACTION_MESSAGE(0x15)	Feedback the check result of connection with server
RESPONE_UDP_INTERACTION_MESSAGE(0x16)	Feedback the check result of the UDP connection
RESPONE_MESSAGE_INTERACTION_MESSAGE(0x17)	Feedback the check result of the SMS connection
RESPONE_APP_INFO_FAIL_MESSAGE(0x18)	Feedback the result indicates the failure to get the application information
RESPONE_APP_RESTRICTION_TIME_MESSAGE(0x19)	Feedback application blocking time information
RESPONE_CONTACT_MESSAGE(0x1c)	Feedback contact information
RESPONE_SMS_BLACK_NAME_MESSAGE(0x1d)	Feedback the black name list of SMS interception
RESPONE_SMS_WHITE_NAME_MESSAGE(0x1e)	Feedback the white name list of SMS interception
SEND_INTERCEPTED_SMS_MESSAGE(0x1f)	Feedback the content of the SMS has intercepted to the control side
RECOVERY_SMS_SUCCESS_MESSAGE(0x21)	Feedback the result indicates the success to recovery SMS

TABLE III.
REQUEST CODES AND MEANINGS

Command code	Meaning
APPLICATION_INTERCEPTION_MESSAGE(0x0)	Add limitation running time span of applications
ADD_ALARM_MESSAGE (0x1)	Set alarm time and message
CHAT_MESSAGE(0x2)	Chat messages
ASK_UDP_INFO_MESSAGE(0x3)	Request UDP socket message
UPDATE_NET_HOOK_KEYWORD_MESSAGE(0x4)	Update the network interceptors keyword message
ASK_GPS_INFO_MESSAGE(0x5)	Request location message
ASK_APP_INFO_MESSAGE(0x7)	Request application information
ASK_FOR_CONNECT_MESSAGE(0xa)	Request to create the connection to the contact
ASK_FOR_APP_NUM_MESSAGE(0xd)	Request the numbers of application
ASK_FOR_CONNECT_SERVER_MESSAGE(0x10)	Request to create connection for server
CHECK_SERVER_INTERACTION_MESSAGE(0x12)	Request to check the connection to server
CHECK_UDP_INTERACTION_MESSAGE(0x13)	Request to check the UDP connection to client
CHECK_MESSAGE_INTERACTION_MESSAGE(0x14)	Request to check the SMS connection to client
ASK_FOR_CONTACT_MESSAGE(0x1a)	Request the contact of the controlled side
ASK_FOR_SMS_NAME_MESSAGE(0x1b)	Request the name list of the SMS interception
RECOVERY_SMS_MESSAGE(0x20)	Request to recovery the SMS which has intercepted
SOS_MESSAGE(0x22)	The controlled side send the message to ask for help to the control side
GENIUS_STOP_WARNNING_MESSAGE(0x23)	The message which indicates the main program of the controlled side has stopped
GENIUS_UNINSTALL_WARNNING_MESSAGE(0x24)	The message which indicates the main program of the controlled side has been uninstalled
GENIUS_DATA_BE_CLEARED_MESSAGE(0x25)	The message which indicates the data of the main program on the controlled side has been cleaned

TABLE IV.
REQUEST CODE / RESPONSE CODE AND MESSAGE TYPE

Request code / Response code	Message Type
APPLICATION_INTERCEPTION_MESSAGE(0x0)	ApplicationRestrictionInfo (custom type)
ADD_ALARM_MESSAGE(0x1)	TipEvent (custom type)
CHAT_MESSAGE(0x2)	ChatMessage (custom type)
UPDATE_NET_HOOK_KEYWORD_MESSAGE(0x4)	String (Java API)
RESPONE_GPS_INFO_MESSAGE(0x6)	GPSInfo (custom type)
RESPONE_APP_INFO_MESSAGE(0x8)	ApplicationRestrictionInfo (custom type)
RESPONE_UDP_INFO_MESSAGE(0x9)	ParentCommunicationInfo (custom type)
ASK_FOR_CONNECT_SERVER_MESSAGE(0x10)	UserInfo (custom type)
RESPONE_APP_NUM_MESSAGE(0xe)	Integer (Java API)
RESPONE_APP_RESTRICTION_TIME_MESSAGE(0x19)	ApplicationRestrictionInfo (custom type)
RESPONE_CONTACT_MESSAGE(0x1c)	ContactContent (custom type)
RESPONE_SMS_BLACK_NAME_MESSAGE(0x1d)	ContactContent (custom type)
RESPONE_SMS_WHITE_NAME_MESSAGE(0x1e)	ContactContent (custom type)
SEND_INTERCEPTED_SMS_MESSAGE(0x1f)	SmsValues (custom type)
RECOVERY_SMS_MESSAGE(0x20)	SmsValues (custom type)

VI. MESSAGE DISPATCHING MECHANISM

More than one module in the system need communicate with the other side. Each module has its own type of communication data. So the system needs to identify and classify the received message according to the message type, and dispatches messages to the corresponding module, and then each module extracts the message and executes instructions [21-22].

The system applies many kinds of communication methods such as SMS communication, UDP communication and TCP communication. Each communication method implements its own InteractionService service, the service is responsible for starting DataReceiver module. The module will start a cycle which is responsible for receiving data, and then starts a concrete receiver - interaction. After the receiver receives the data, the data will be submitted to the InteractionMessage module to analysis and format. When the message is returned to the DataReceiver, DataReceiver checks its validity, then the effective data will be submitted ExecuteInteractionMessage to the execution module -, the dispatch module will send the message to the responding module by the message type, and the responding module will execute specific instruction to complete a process of receiving and parsing

the communication data. Figure 4 shows the message dispatching sequence diagram.

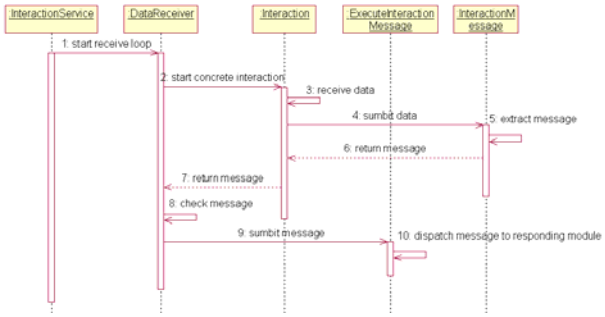

Figure 4 Message dispatching sequence diagram.

VII. The System Implementation

A. Control Side Structure

The control side can manage and control the controlled side. Taking into account actual situation, the control side is designed with one-to-many control mode. Control functions provided by the control side and the monitoring modules of controlled side are one-to-one correspondence, respectively, GPS monitoring, alarm event setting, application running time monitor, network interception monitor, the SMS interception set and end-to-end chatting. Before running the above modules, the program of control side will let the user select the contacts which he wants to control. The structural diagram of the control side is shown in Figure 5.

B. The Controlled Side Structure

The Controlled side is divided into two parts, the main program and the saver. When the main program opened, it will automatically start the saver. The main program is

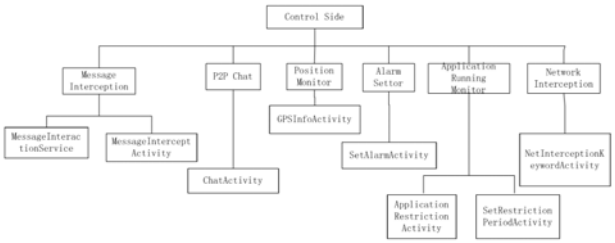

Figure 5 The structural diagram of the control side.

responsible for running online behavior management monitoring module, the saver is responsible for monitoring the running status of the main program. When the main program closes unexpectedly or is uninstalled, the saver will access the contact information through a shared file of the main program, and notify the running status timely to the control side program. The interaction among the two clients and the saver is shown in Figure 6.

C. Alarm Setting Implementation

As a brief description of the one-to-many control mode on the control side, we show how to set alarm settings. In the control side, users can set the alarm clock event on the controlled side. When setting the alarm event, users firstly select the contact who they want to add an alarm to,

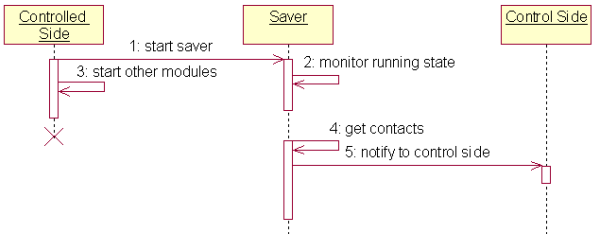

Figure 6 The interaction among the two clients and the saver.

select the alarm time and set reminding text. The system will issue the instruction to add the alarm after confirmation. Setting the alarm event sequence diagram on the control side is shown in Figure 7.

The controlled side will perform some operations such as adding alarm after receiving the instruction. The message dispatching module will dispatch the message to the TipEventManager module, this module will add an alarm event which described in the message, and update the alarm events list in the GeniusAlarmManager module.

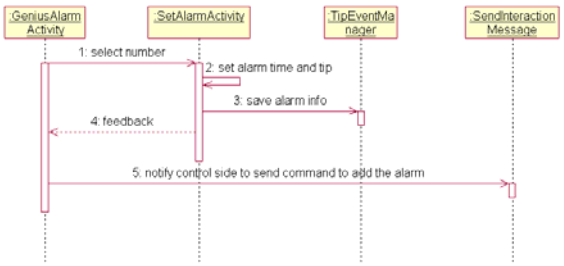

Figure 7 Setting the alarm clock event sequence diagram on the control side.

The GeniusAlarmManager module will add alarm intent to CallGeniusAlarm module which can be woke up on time by Android system, so that the user can get a alarm on the particular moment. At last, TipEventManager module updates the new alarm events on the user interface. Setting alarm clock events sequence diagram on the controlled side is shown in Figure 8.

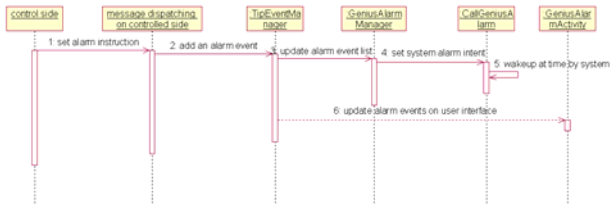

Figure 8 Setting alarm clock events sequence diagram on the controlled side.

D. Integration with GIS

The control side can monitor the real-time position of the controlled side. First of all, the user interface class GPSInfoActivity will send a request for monitoring the position of the controlled side. After the controlled side receives the request, it will get its real-time position location information by GPS or network, and then return the message to the control side. When the control side receives the message which includes the GPS information, the message will be dispatched to the GPSInfoManager module. GPSInfoManager module will update the list of messages, and query the corresponding location name by using latitude and longitude in the GPS message. At last, GPSInfoActivity will be notified to initialize the map view and display the location of the controlled side on the map. The GPS request processing sequence diagram on the control side is shown in Figure 9.

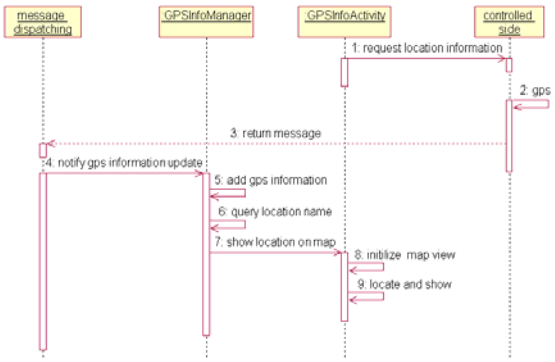

Figure 9 GPS request processing sequence diagram on the control side.

The GIS module uses the map API of the Baidu Company to locate and display the GPS information on the map. Firstly, some initialization operations are executed by using the GPS information and the system create a GeoPoint object to add to the CustomItemizedOverlay. The overlay calls the MapView to update its view, so that the map view calls its MapController to complete the display work. The GIS displaying sequence diagram is shown in Figure 10.

When the controlled side receives the GPS locating request, the message dispatching module dispatches the message to the GPS module. The LocationManager in GPS module gets the location manager of Android System to initialize itself, and initializes a location listener. The listener will monitor the mobile's location change. When the listener gets concrete location information, it will create a GPSInfo object by this

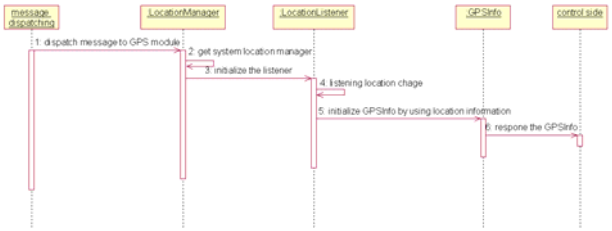

Figure 10 GPS request processing sequence diagram on the controlled side.

location information and send this message to the control side. The GPS request processing sequence diagram on the controlled side is shown in Figure 11.

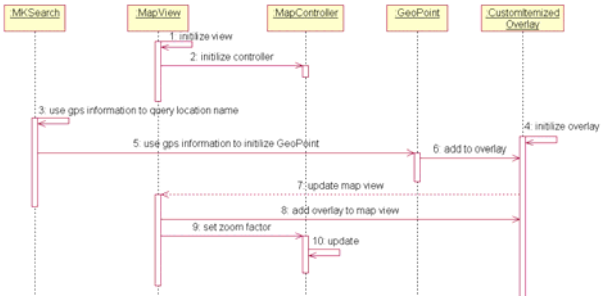

Figure 11 GIS displaying sequence diagram on the control side.

E. Network Interception Module

In this module of controlled side, coding the network interception driver is difficult, because of the security reasons on Android. The platform does not provide network interception in its API. Android operating system is written based on Linux; the kernel provides a framework-NetFilter in the underlying Linux. Using this framework, the problem of network interception modules can be solved [22-23].

NetFilter is a scalable framework of the Linux kernel; it allows developers to define the hook function in the middle of the Linux network protocol layers. When network data flow through the layers of the protocol layer in the kernel, the kernel will automatically call the appropriate hook function that specific statement and return value which indicates the packet processing results: through, discarded or taken over. Hook functions can be registered at any protocol layer in the kernel and listens the data flow, then return value NF_ACCEPT (go through, to pass on), NF_DROP (discard, no longer pass) and so on.

The network interception module must be running in the Android kernel, but the system module is running in the upper layer of Android users, so they must have an interactive mechanism that makes other modules set and read the data of the network intercept module. Some operations of the interception module must require other modules to complete some work, such as the network interception keywords setting, installing and uninstalling module. So, in order to interact with other modules in convenience, the network module uses two files which

are in the specific path. One is a log file recording the keyword interception history of the module, the other file is a keyword list file, which is a collection of keywords that module needs to intercept them. When other modules need to change the keyword, they only simply write the new keywords to this file, and reset the keyword of the network interception module. If other modules want to get the interception history, they only simply access the log file.

During the interception process, the module reads the current data packet to judge that whether it contains the words which are needed to intercept, and then judges that whether it is a TCP packet to process further, otherwise return NF_ACCEPT to let the data go through. Because under normal circumstances, keyword searching function is completed in TCP request, so the system only needs intercepting TCP packets to be processed.

After the TCP packet is intercepted, the system extracts the contents of the TCP packet and matches the keywords in the keyword table. If the match is successful, we believe that the message contains sensitive content and should discard this packet. A matching algorithm - KMP algorithm is used in the matching process. The algorithm limits the running time in linear time, this can improve the response time of the kernel module, and also be a great help to optimize the performance.

Network interception workflow on the controlled side is shown in Figure 12. The network module has a network interception services unit; this unit will start the service when the program starts. After the start of the service, the network interception unit will be installed; this unit is the core of the entire module. After installing the unit, the system will launch network interception. The interception unit will start an interception log unit, this unit need to update the interception keyword list, so it will issue the request of the update keywords to the keyword Manager. The module also has a unit used to display an interface to show the search keywords, the interface will display by the trigger of users.

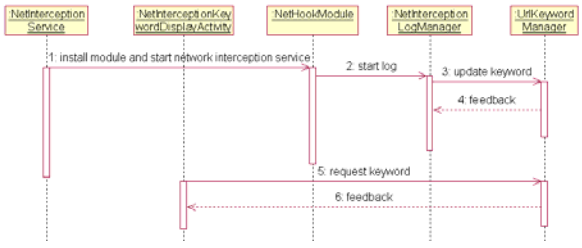

Figure 12 Network interception sequence diagram on the controlled side.

On the control side, users edit keywords of network interception. Users call the UI-NetInterceptionKeywordActivity first; the user interface class requests the recent keywords by calling the keyword manager-NetInterceptedKeywordManager. Users edit the keywords on UI. The NetInterceptionKeywordActivity updates the keywords in the manager, and the manager sends a request of keyword updating to the controlled side to complete keywords synchronization operation. The keyword synchronization sequence diagram on the control side is shown in Figure 13.

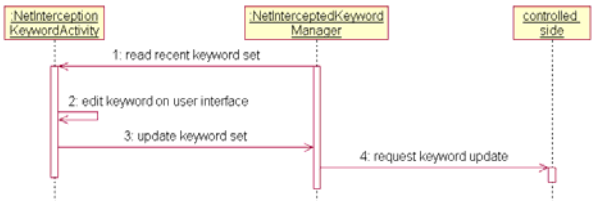

Figure 13 Keyword sequence diagram synchronization on the control side.

F. SMS Interception

On the controlled side, SMS interception module registers a SMS listener, and realizes the SMS interception list management, contacts management and SMS backup functionality. When an SMS broadcast reaching, the listener will automatically be invoked by the system. The listener will extract the number of the SMS sender to submit to the interception name list management module, this interception name list management module requests the phone contacts to contacts manager and updates stranger List and interception name list. If the number is on the blacklist or a stranger, the intercept list manager returns the interception response to the listener. Received by the response of interception, the listener will notify the SMS backup module to back up the SMS which is going to intercept, and the broadcast is terminated. The SMS interception sequence diagram on the controlled side is shown in Figure 14[24-25].

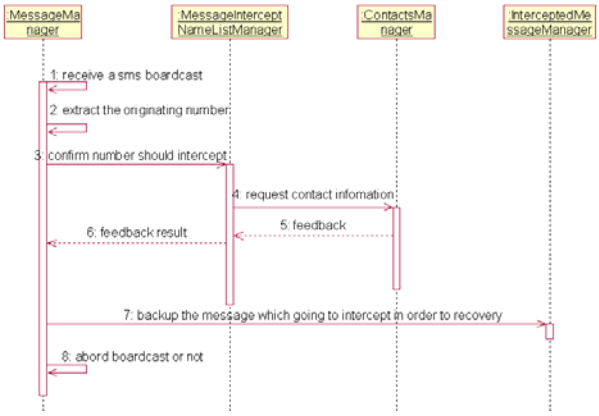

Figure 14 SMS interception sequence diagram on the controlled side.

On the control side, users edit the name list of SMS interception and are notified when a SMS is intercepted by the controlled side. The user interface class-MessageInterceptActivity requests the name list for a manager-MessageInterceptNameListManager and shows the name list on UI. If the name list is empty, the manager will send a message to the controlled side which requests the contacts information of the mobile. When users edit the name list, the program updates the data of the manager and notifies the controlled side to update its

name list in time. When the controlled side intercepts a SMS, the system notifies a manager-InterceptedMessageManager on the control side, and it notifies users finally. The SMS interception sequence diagram on the control side is shown in Figure 15.

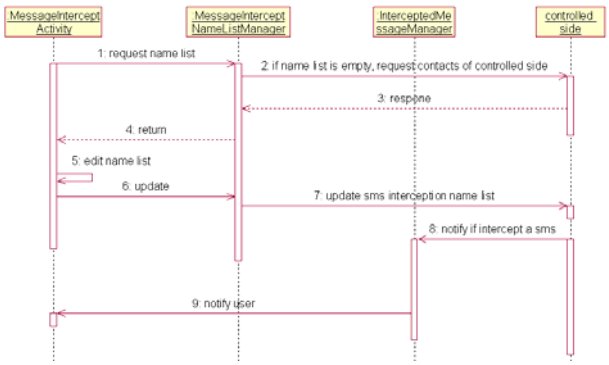

Figure 15 SMS interception sequence diagram on the control side.

G. Application Control

In order to achieve monitoring applications and limit running time of applications, the program of controlled side implements a Linux command module which is running on Android platform. This module uses commands to manage and limit the running time of applications. For behavior management software, which is a very important breakthrough, because Android has a series of security mechanisms, the Android top level API function never allows to force to shut down the top-level program, so the program only can monitor and manage the running time of the applications by using Android underlying mechanism.

Application control flow is shown in Figure 16.

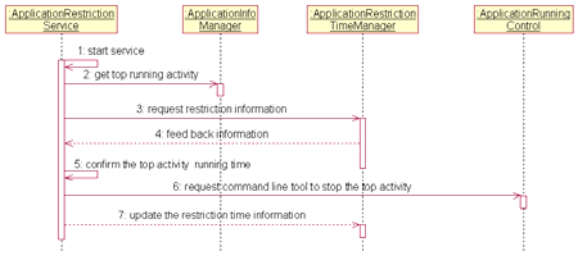

Figure 16 Application control sequence diagram on the controlled side.

Application control services unit will be started at first, this service will get the information of applications at the top of phone screen regularly through the application information manager. Then it will request applications run time information to the application run-time manager by sending applications' information. Service unit will further confirm whether the run time of top-level applications exceeds the pre-assigned run time. If that is true, the service unit will request the command line control tool to shut down the top-level program, and gives a user prompt. Finally, the application run-time manager will update the application run-time information.

On the control side, users edit the application restriction information on the user interface. The interface class-ApplicationRestrictionActivity requests the application information for the manager-ApplicationRestrictionTimeManager. If the data is empty, the manager requests the application for the controlled side, and then updates the user interface. When users select an application name, the interface jumps to the other interface-RestrictionApplicationTimeActivity. users can modify the restriction time span or restriction instants. The manager is notified and sends a message to update the application restriction information on the controlled side. The setting application restriction information sequence diagram is shown in Figure 17.

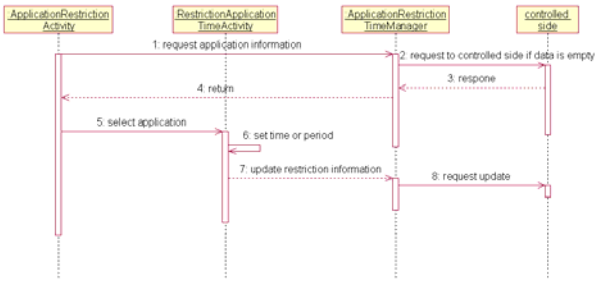

Figure 17 Setting application restriction information sequence diagram.

H. Server Side Implementation

The server side mainly controls the client side by online user management. It acts like an information transfer and processing media to achieve a reliable remote control function.

The interaction among the server and the clients is shown in Figure 18.

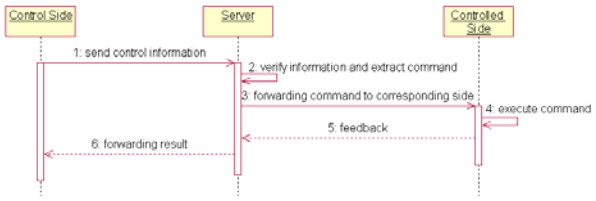

Figure 18 The interaction among the server and the clients.

The control side sends the control information to the server, the server verifies the identity information, and finds the corresponding controlled side and send the control information to it. The controlled side will fetch instruction according to the control information, and then perform the appropriate action. If the control side needs feedback of any action, such as GPS cannot get the location information sometimes because of the network problems, the controlled side needs to send the appropriate message to the control side, informs the control side that the action executes failed, and then the control side can take some appropriate action to solve problems.

G. User Interface Implementation

The control side and the controlled side have different functions, but the design of their user interface is similar, that can support the consistency of the user experience. The design of user interface is simple and clear, so that the user can get a better experience when using this software. The figure 19 and the figure 20 is some UI screenshots of this software.

Figure 19 The start interface of the control side.

Figure 20 The start interface of the controlled side.

VIII. CONCLUSIONS

The Internet behavior management system provides a good functional support for parents to monitor children's mobile phones. The system is designed within Android kernel deeply, and already implements the functions that the Android API cannot be completed, such as network data filtering, monitoring running time of applications, and also makes the software's performance be optimized on Android platform. To improve the functionality of the system, provide a complete server support and strengthen user authentication for security mechanisms will be the further research directions of the system.

ACKNOWLEDGMENT

The authors wish to thank Haojie Zhang, Xiaorui Xu, Zhongcheng Feng and Jihao Chen. They did a lot of work during the system testing. This work was supported in part by the National Natural Science Foundation of China under Grant No. 11271003 and Programs in higher school high-level talents of Guangdong Province.

REFERENCES

[1] Wookey Lee, Woong-Kee Loh, Mye M. Sohn, Searching Steiner trees for web graph query. *Computers and Industrial Engineering.* 62h ed., vol. 3, pp. 732-739, 2012.
[2] Murat Gunestas, *An evidence management model for web services behavior*. Fairfax, VA USA: George Mason University; 2009.
[3] Hayato Ohmura, Teruaki Kitasuka, and Masayoshi Aritsugi, Web browsing behavior recording system. Proceedings of the 15th international conference. In *Berlin*, Heidelberg, 2011, pp. 53-62.
[4] Guo Danhua, Bhuyan Laxmi Narayan, Liu Bin, An Efficient Parallelized L7-Filter Design for Multicore Servers. *IEEE-ACM TRANSACTIONS ON NETWORKING.* Vol. 20, No. 5, pp. 1426-1439, 2012.
[5] Garcia Kunzel, Adriana. An Android approach to the web services resource framework. Florida Atlantic University. ComputerScience. 2010.
[6] Mohammad Nauman, Sohail Khan, and Xinwen Zhang, Apex: extending Android permission model and enforcement with user-defined runtime constraints. Proceedings of the 5th ACM Symposium. Suite 701 New York NY USA, 2010, pp. 328-332.
[7] Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner, Android permissions: user attention, comprehension, and behavior. Proceedings of the Eighth Symposium. Suite 701 New York NY USA, 2012, pp. 1 -14.
[8] Bhaskar Pratim Sarma, Ninghui Li, Chris Gates, Rahul Potharaju, Cristina, Nita-Rotaru, and Ian Molloy, Android permissions: a perspective combining risks and benefits. Proceedings of the Eighth Symposium. Suite 701 New York NY USA, 2012, pp. 13-22.
[9] Alexander Sirotkin, The Java API to Android's telephony stack. *Linux Journal.* 2009th ed., vol. 183, pp. 1, 2009.
[10] Onur Cinar, *Android Apps with Eclipse. First Edition*. CA USA: 901 Grayson Street Suite 204 Berkely, 2012, pp. 20-21.
[11] Marko Gargenta, *Learning Android*. CA USA: O'Reilly Media Inc, 2011, pp. 28-32.
[12] Victor Matos, and Rebecca Grasser. Building applications for the Android OS mobile platform: a primer and course materials. *Journal of Computing Sciences in Colleges.* 26th ed., vol. 1, pp. 23-29, 2010.
[13] Jeff Friesen, *Learn Java for Android Development.* CA USA: 901 Grayson Street Suite 204 Berkely, 2010, pp. 65-66.
[14] Min Zhao, Tao Zhang, Fangbin Ge, and Zhijian Yuan, RobotDroid: A Lightweight Malware Detection Framework On Smartphones. *Journal of Networks.* Vol. 7, No. 4, 2012.
[15] Min Zhao, Tao Zhang, Jinshuang Wang, and Zhijian Yuan, A Smartphone Malware Detection Framework Based on Artificial Immunology. *Journal of Networks.* Vol. 8, No. 2, 2013.
[16] Zhizhong Wu, Xuehai Zhou, and Jun Xu, A Result Fusion based Distributed Anomaly Detection System for Android Smartphones. *Journal of Networks.* Vol. 8, No. 2, 2013.
[17] Lee Boon-Giin, Chung Wan-Young, A Smartphone-Based Driver Safety Monitoring System Using Data Fusion. *SENSORS.* Vol. 12, No. 12, pp. 17536-17552, 2012.
[18] Payet Etienne, and Spoto Fausto, Static analysis of Android programs. *INFORMATION AND SOFTWARE TECHNOLOGY.* Vol. 54, No. 11, pp. 1192-1201, 2012.
[19] Zhang Ying, Huang Gang, Liu Xuanzhe, Zhang Wei, Mei Hong, Yang Shunxiang, Refactoring Android Java Code

for On-Demand Computation Offloading. *ACM SIGPLAN NOTICES*. Vol. 47, No. 10, pp. 233-247, 2012.
[20] Njunjic, Ivan. Development Techniques for Android Platform Mobile Device Application. Eastern Michigan University.bComputer Information Systems. ComputerScience. 2012.
[21] Lombera, Moser, Melliar-Smith, and Chuang, Mobile Decentralized Search and Retrieval Using SMS and HTTP. *MOBILE NETWORKS & APPLICATIONS.* Vol. 18, No. 1, pp. 22-41, 2013.
[22] Brown Anthony, Mortier Richard, Rodden Tom. MultiNet: Usable and Secure WiFi Device Association. *COMPUTER COMMUNICATION REVIEW*. Vol. 42, No. 4, pp. 275-276, 2012.
[23] Elisa Bertino, Elena Ferrari, and Andrea PeregoA, General Framework for Web Content Filtering. *World Wide Web.* 13th ed., vol. 3, pp. 215-249, 2010.
[24] Ruining Huang, Lei Li, Yunjiang Lou, Research and Construction the Net Monitor System, Special Issue: Advances in Information and Networks. *Journal of Networks*. Vol. 7, No. 7, 2012.
[25] Nuruzzaman M. Taufiq, Lee Changmoo, bin Abdullah Mohd. Fikri Azli, Choi Deokjai, Simple SMS spam filtering on independent mobile phone. *SECURITY AND COMMUNICATION NETWORKS.* Vol. 5, No. 10, pp. 1209-1220, 2012.

Miao Liu was born in Hubei Province, China on September 24, 1969, and received the B.Sc. degree, the M.Sc. degree in Computer Science from the Information Engineering University, China, the PhD degree in Computer Application Technologies from South China University of Technology China, in 1987, 1991, and 2007, respectively.

He is currently an associate professor at the computer science department of Guangzhou University, in Guangzhou, China, and has authored and co-authored over 30 technical papers. His major research interests are: network security, artificial intelligence and e-commerce.

Gengtong Hou was born in Guangdong Province, China on February 24, 1990, and is studying in Guangzhou University, China.

Ying Gao was born in Hubei Province, China on June 1,1963, and received the B.Sc. degree in mathematics education from the Central China Normal University, China, and the M.Sc. degrees in computer science from Beijing University of Aeronautics and Astronautics and the PhD degrees in Communication and Information Systems from South China University of Technology China, in 1987, 1998, and 2002, respectively.

He is currently a professor at the computer science department of Guangzhou University, in Guangzhou, China, and has authored and co-authored over 60 technical papers. His major research interests are: evolutionary multiobjective optimization, constraint-handling techniques for evolutionary algorithms.

Prof. Gao is a member of the IEEE Guangzhou subsection.

Chunming Tang was born on Jan 1, 1972, in Hunan Province, China, and received B. Sc. Degree in mathematics education from Xiangtan Normal University, China, and the M.Sc. degree in Computational Mathematics from Xiangtan University, China, and PhD degree in Applied Mathematics from Chinese Academy of Science, China.

He is a mathematics professor in Guangzhou University. His major research interests are Cryptography and Cloud Computing.

Dongqing Xie was born in 1965, and received his PhD degree from Hunan University in 1999.

He is currently a professor and PhD supervisor at the computer science department of Guangzhou University, China. His main research interests include algorithm analysis and design, information security. He is also a member of China Computer Federation.