Design and Implementation of anti-cheating programming Examination system

Yiju Wei Hong Han Jianyua Jiang Yong Liao

College of Computer Science and Engineering
University of Electronic Science and Technology of China, Chengdu, 610054
weiyiju@live.cn

Abstract

Programming oriented on-machine examination (POME) faces some new cheating challenges. They exploit the features of third-party programming tools used in POME. Traditional methods, such as screen taking over, packet filtering, used in traditional on-machine examination system could not prevent cheating based on third-party tools, so they are useless in POME. To address the new cheating problems in POME, an API hooking based hybrid mechanism is proposed. By killing processes in a black list and hooking cheating related system APIs in third-party programming tools, the system can effectively decrease cheating rates in a POME

Keywords: anti-cheating examination system, API hooking, Process killing

1. Introduction

There are two kinds of on-machine examinations: traditional on-machine examinations and **programming oriented on-machine examination** (POME). Traditional on-machine examinations only require examinees to fill answers of the questions. To prevent cheating, several strategies are proposed [1-5]. For example, Processes that are not necessary for examination would be killed [1]. All packets that are not sent to server would be discarded [2]. By changing configuration of the operating system, examinee's operations on computer would be restricted [3]. Random paper generation algorithms are used to give each examinee different paper [5].

POME requires examinees to build programs with third-party development tools like visual c++. These strategies do not solve the problems resulting from utilization of third-party tools. For example, killing processes cannot be applied to third-party tools, because the processes of the tools are necessary to build programs. In addition, examinees can build a program with third-party tools to kill the process of examination system itself which is responsible for killing illegal processes. Because in a POME the cheating actions involve local operations, for instance, examinee can save hints on local computer to help examinees who use the same computer in next turn of examination, the packet discarding strategy for network is not suitable. Traditional examination systems can restrict examinee's operations on the computer. But examinees in a POME can use third-party tools to build a program to execute a system command to bypass the restrictions. Additionally, random paper generation algorithms is not proper also, in a POME, the test paper must be the same for all examinees. In conclusion the anti-cheating strategies for traditional examination system are invalid for POME. Also, related works for POME have not been seen.

The system described in this paper is designed for POME. It gives an examination interface to examinees and allows them to use third-party tools. But some functions of third-party tools are restricted by system.

In section 2, possible cheating methods to an on-machine examination is analyzed. In section 3, the system structure and defense mechanism of these cheating methods are presented.

2. Cheating methods Analysis

In this section, the possible cheating methods in an actual on-machine examination are analyzed.

Assume that Examinees sit in a classroom with an individual computer on each desk. Except talking or moving, they can do anything on their own computer. These computers are connected to a LAN through which answers are sent to server. Third-party tools like visual studio or vc6.0 are installed on each computer to build examinee's project. Examinees are divided into several groups and examination is divided into several turns. A group of examinees get into the classroom in each turn while other examinees wait outside.

There are two kinds of possible cheating methods: the common and special methods. Common cheating methods can be used in any kind of on-machine examinations. Special cheating methods can only be used in a POME.

2.1. Common cheating methods

These methods are as follows:

a. Exchange files with others via network tools, such as telnet, ftp and so on.

b. Use task manager to kill the processes of the examination system. In windows, by pressing ctrl+alt+del, this could be achieved.

c. Use explorer to open a file which was copied there before the examination

d. Use a USB storage device to copy relative materials.

e. In last turn of examination, examinees copy his code to the clipboard. Then others in next turn could take advantage of it.

2.2. Special cheating methods

These methods can only be used in a POME:

f. With third-party development tools (not operating common component like explorer), project files can be saved another copy in some directories, and other examinees can use it in next turn of examination. For example, if 'save as..' button is clicked, IDE will popping a file window to ask user to give a new directory. Another way to save a project file to some directory is create a new project in that directory and click 'save' button,

g. Although without the help of explorer, files could be copied to certain directory using open file window of third-party tools. After that, others could open them.

h. Examinee can build a program with IDE to invoke library functions like 'system' to execute a shell command to copy a file or open an existed file.

i. Examinee leaves his project in a system directory which is allowed to use and others can open it in next turn.

j. With IDE, examinees can build a program to kill the process of examination system and cheat. Invoking functions like TerminateProcess could achieve it.

3. Design and implementation of the system

3.1. System overview

When an examinee begins examination with the system, an ID and a password is required to verify the examinee’s identity. The ID and password will be sent to a server, and after the server verified the ID matches the password, the server will send examination related materials to the system. Then the system takes over the whole user interface, through which only designated third-party tools could be used to build program. Furthermore, examinee cannot use the third-party tools to cheat because some functions of the third-party tools are disabled. When the examinee finished the examination, the answers will be sent to server.

3.2. System structure

The system includes four modules: third-party tools monitoring module, processes monitoring module, assignment manager module and the user interface module. The processes monitoring module monitors and kills some forbidden processes. Third-party tools monitoring module contains a hooks library for the third-party tools. It is implemented with detours [6]. The hooks library is built as a DLL (dynamic linking library). This DLL will be injecting to third-party tools and disable some original functions. The assignment manager module is used to connect and exchange data with a server. The data includes the user information, examination related materials and project files and so on. The user interface module is implemented with Qt library. It includes two windows: login window and examination window. A configure file that can be edited by administrator to configure the server IP address, the directory of third-party tools and what processes should be killed.

Figure 1 shows the structure of the system.

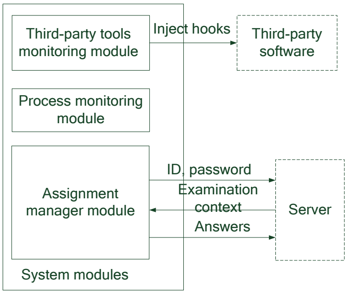

Fig.1: System structure

3.3. Defense mechanism

To defense cheating methods described above, the system will try to kill some processes. Additionally, the system uses detours to hook APIs to defend special cheating methods. Moreover there’re also some other methods used.

To defense cheating method a, b and c, the system try to kill processes like explorer, taskmgr, cmd and processes related to telnet or ftp. Teachers could decide what processes should be killed by editing a configure file.

To defense cheating method f and g we hooked file operation related system APIs:

CopyFile,
CreateFile,
GetOpenFileName,
GetOpenFileNameA,
GetSaveFileName,
GetSaveFileNameA,
SHBrowseForFolder.

The system makes these APIs returning directly without doing anything.

When a new process is created, the hooks should be inject into it to make sure examinees cannot invoke forbidden functions .The system hooks the ‘CreateProcess’ function to make sure every processes cannot use APIs above.

The system do not allowed examinees to edit Registry. So the system hooked the registry operation related APIs:

RegCreateKeyEx,
RegSetValueEx.

To defense cheating method h, the system hooked ‘system’ C library function. When examinee debugs a code that contains invoking of ‘system’, the ‘CreateProcess’ hook will disable it without doing anything.

To defense cheating methods i, the system will clean project directory after an examinee make sure that he has finished the examination.

To defense cheating method d, the system disables USB devices in two ways: by changing key values in registry of windows ("Software\\Microsoft\\Windows\\CurrentVersion\\Policies\\Explorer"), or by invoking ‘SetupDiRemoveDevice’ function when a device is connected.

To defense cheating method e, the system cleans clipboard by invoking ‘EmptyClipboard’ function after an examinee finished his or her examination.

The relationship between defense methods and cheating methods is shown in figure 2.

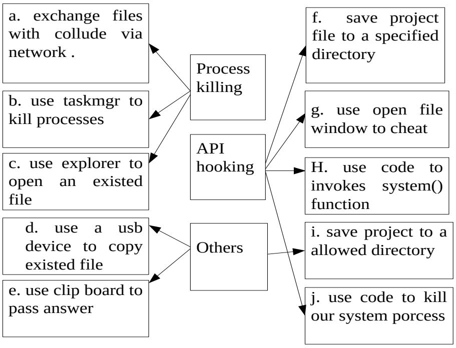
a. exchange files with collude via network .
b. use taskmgr to kill processes
c. use explorer to open an existed file
d. use a usb device to copy existed file
e. use clip board to pass answer
Process killing
API hooking
Others
f. save project file to a specified directory
g. use open file window to cheat
H. use code to invokes system() function
i. save project to a allowed directory
j. use code to kill our system porcess

Fig. 2: Defense mechanism

4. Conclusion

This paper presented an examination system for POME. It not only defense common cheating methods in common examinations but also defense special cheating methods in POME by hooking APIs of the third-party tools that is used in POME. The system can guarantee the fairness of the POME. The system has a good performance in POME in UESTC (university of electronic and technology of China).

5. References

[1] Guo Dong, Jia Man Lei, Wang Xu Wan “The Solution To Prevent Examine From Cheatlng Of Paperless Examinatlon System” In Journal Of Nanyang Institute Of Technology [J], VO1．5 NO．5 pp.21-23, Mar 2010.

[2] WU WEI, WEI Xia, WEI SHI MIN “Research of defending online-test cheating based on manager server” In Computer Engineering and Design [J], VO1．28 NO．8 pp.1941-1943 Apr．2007.

[3] XU Qiao-zhi, LIU Dong-sheng “Research and Design of the Security of Network Examination System”. Computer Education [J], VO1．28 NO．8 pp.40-41, No.5 Mar.10, 2010.

[4] TAO Xia， CHEN Hong—liang “Design and Realization of Online Examination System for Circuit”. RESEARCH AND EXPLORATlON IN LABORATORY [J], VO1．27 NO．7 pp.83-86 Jul．2008.

[5] ZENG Yi, RAN Zhong, GUO Yong-lin “Algorithms of automatically generating test paper in database and strategies of measuring paper”. Computer Engineering and Design [J], pp.3024-3026, VO1．27 NO．16 Aug. 2006.

[6] Galen Hunt and Doug Brubacher “Detours: Binary Interception of Win32 Functions” In Third USENIX Windows NT Symposium.