Advanced Administration of Windows Based on Open Source Utilities

Ademi Ospanova, *Member, IAENG*, Berik Tuleuov, Gulzat Karzhauova, and Lazzat Kussepova

***Abstract*— A set of works aimed at optimizing certain administrative processes of Windows operating systems is done. A classification of administration tasks has been developed in order to determine both labor-intensive and routine administration tasks, as well as tasks requiring and allowing automation. Based on preselected criteria a comparative analysis of administration tools, including built-in and open source utilities is made. A plugin for the Process Hacker utility, which allows creating and control of the list of allowed and conditionally prohibited programs in Windows 7 SP1, Windows 8, Windows 10 operating systems has been developed and deployed.**

***Index Terms*—Windows, open source, Process Hacker, system management services, low-level programming, file system driver**

I. Introduction and the Problem Statement

A large percentage of the PCs (workstations) used nowadays in organizations runs Windows operating systems. The level of qualification of a system administrator, as well as the tools used to control and configure operating systems, corporate networks, and peripheral devices, are largely determining factors for the security, fault tolerance, and reliability of an organization's infrastructure. Built-in administration tools are not always convenient for effective management of system processes, and may not possess configurable capabilities for performing both routine and special administration tasks ([1]). There are many third-party specialized utility programs, both open source and commercial ([2]-[5]). However, the specific tasks of administration and security, the emergence of new technologies give rise to new administrative tasks and cause the need for their automation. At the same time, there are requirements for the system administrator to be acknowledged with scripting languages, also to be able to work with programs with a command-line interface ([6]-[8]).

Manuscript received March 17, 2021; revised April 07, 2021. This research is funded by the Science Committee of the Ministry of Education and Science of the Republic of Kazakhstan (Grant No. AP09561712).

A. Ospanova is Associated Professor of Information Security Department, L. N. Gumilyov Eurasian National University, Nur-Sultan, Z01A3D7 Republic of Kazakhstan (e-mail: o.ademi111@gmail.com).

B. Tuleuov is Senior Lecturer of Mathematical and Computing Modeling Department, L. N. Gumilyov Eurasian National University, Nur-Sultan, Z01A3D7 Republic of Kazakhstan (e-mail: berik_t@yahoo.com).

G. Karzhauova is Master Student of Information Security Department, L. N. Gumilyov Eurasian National University, Nur-Sultan, Z01A3D7 Republic of Kazakhstan (e-mail: o.ademi111@gmail.com).

L. Kussepova is PhD Student of Information Security Department, L. N. Gumilyov Eurasian National University, Nur-Sultan, Z01A3D7 Republic of Kazakhstan (e-mail: o.ademi111@gmail.com).

The problem of creating personalized tools for advanced administration of operating systems can be reduced to solving the following main tasks:

1. Studying of the Windows device, as well as API tools from the point of view of system programming in order to create utilities and drivers with enhanced privileges ([1], [9]-[10]);
2. Setting, algorithmizing, and implementation administrative tasks that require automation ([4]-[5], [12]-[15]).

In the course of the work, it is necessary to develop a classification of the tasks of the operating system administration according to the criteria of basicity/peculiarity, routine and automation capabilities, existing software tools for these tasks. It is also necessary to conduct a comparative analysis of some system service management utilities, defining the comparison criteria. Ultimately, it is necessary to solving the problem of developing a plugin for Process Hacker with the required functionality.

Types and methods of the conducted research:

- Research and analysis (classification of administrative tasks, determination of the actual task, comparative analysis of utilities and built-in administrative tools);
- Programming (development of a plugin with an automatic launch for the Process Hacker utility, capable to create and control lists of allowed and conditionally prohibited programs in the system);
- Debugging and testing the driver and the plugin in different versions of operating systems, taking into account the Windows policies.

II. Windows Administration Tools and Instruments

System administration of Windows operating systems includes the following tasks:

- Installing, configuring, and updating various types of software;
- Connecting and configuring peripherals;
- Launching and configuring system services (system configuration);
- User account management, technical support;
- Process Management;
- Resource allocation;
- Monitoring of the system operation, ensuring information security, fault tolerance;
- Providing data backup, performing data recovery.

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

A. *Windows Built-in Windows Administration Tools*

In later versions of Windows, there are built-in tools for administration and computer management – the "Administration" section. Windows 10, 8.1, and Windows 7 are well-endowed with useful built-in system utilities:

- "System configuration". Allowing to configure the operating system boot settings.
- "System Information".
- "Troubleshooting Windows".
- "Computer Management". Provides tools such as:
  - Task Scheduler;
  - View Windows events;
  - Resource Monitor;
  - Disk Management;
  - System Stability Monitor;
  - Disk cleanup;
  - Windows Memory checker.
- "Task Manager".

B. *Third-Party Utilities for the Operating System Administration*

In the work the following utilities were considered:

- Advanced IP Scanner – Local network scanner,
- NetWrix Inactive Users Tracker – search for inactive accounts,
- WinAudit Freeware – Operating system audit,
- Performance Analysis of Logs – system performance analysis,
- Free Active Directory Tools – free analogue of Active Directory,
- Comodo Time Machine – rollback of the operating system,
- System Explorer – monitoring running processes and services, checking for threats and rootkits,
- Daphne – monitoring the system operation, stopping processes on a schedule,
- MiTeC Task Manager – monitoring running processes and services, viewing open files, launch dates,
- Process Explorer (a utility from Microsoft's SysInternals) – monitoring of running processes and services, disks, and processor operation,
- Process Hacker – free and open-source program that provides considerable administration capabilities for Windows operating systems, while there is an API for developing personalized plugins.

Table I provides a brief comparative analysis of the Process Explorer and Process Hacker utilities.

TABLE I
COMPARISON OF THE PROCESS EXPLORER AND THE PROCESS HACKER UTILITIES

Criteria	Common characteristic	Process Hacker	Process Explorer
Installation	Both programs have a portable version	easy to launch and run	Acceptance of the terms and conditions is required
Checking for updates		included	not applicable
Tray icons	Both programs provide the utility	By default, shows the CPU load in both User Mode and Kernel Mode. Up to 8 tray icons with different information can be enabled	By default, shows only the CPU load in User Mode. Up to 7 tray icons with different information can be enabled
Notification of processes\services\drivers		Start\stop\installation notifications of the services and drivers are provided	not applicable
Interfaces	The program interfaces are similar, conveniently displaying the process tree		
Coloring	Coloring is available in both programs	By line	By column
Filter by process name		Included, supports keywords to search for certain types of processes	not applicable
Performance charts in the toolbar		not applicable	Included
Services and drivers		Works with services and drivers	Works only with normal processes
Modular architecture		Modular, supports plugins (a significant part of the functionality is implemented by plugins)	One-piece
Priority		The priority of actions can be set	Affinity and Priority actions for processes can be set
Adding functionality		easy to launch and run	not applicable
Coloring		included	
Filter by process name		Possible, the API is provided	

III. DEVELOPMENT OF THE PROCESSDEEPMANAGE PLUGIN

A. *Process Hacker and Plugins creation*

The source code of Process Hacker 3.0, the latest version at the moment of writing, is available on the GitHub resource ([16]), the source code of additional plugins – ([17]). To build, modify, and debug the utility, Installation of the SDK and the WDK packages is needed. The Software Development Kit (SDK) is a set of tools, code samples, documentation, compilers, headers, and libraries that developers can use to create applications running on Microsoft Windows operating systems. Windows Driver Kit

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

(WDK) is a software toolset from Microsoft that allows the development of device drivers for the Microsoft Windows platform. Building tools – MSBuild, Visual Studio 2019; build in Release mode x64 bit rate ([18]-[19]). There were linking errors, and the Terminator plugin project had to be excluded from the solution. To enable additional plugins, making changes to the configuration file 'ProcessHacker.exe.settings.xml', located in the \bin\Release64 directory is required ([20]).

B. *Development of the ProcessDeepManage plugin*

Fig. 1 illustrates the code snippets for embedding the plugin in the process Hacker utility interface. The plugin interface is designed in the Kazakh language. Fig. 2-3 demonstrate a working plugin that is available from the context menu and in the application tabs.

C. *Development of a driver for the plugin*

The driver was created on the base of the file system filter driver. The functionality of the driver described in Table II.

TABLE II
DRIVER FUNCTIONALITY FOR THE PROCESSDEEPMANAGE PLUGIN

	Action	Name of menu items
The driver's capabilities for managing operating system processes and the corresponding menu items	Disable startup	Орындауға тиым салу
	Add to the prohibited list	Қара тізімге енгізу
	Add to the list of conditionally allowed users	Шартты тізімге енгізу

Starting with versions older than Windows Vista, system programming and integration of the developed software have some specific difficulties. For example, in Windows 7, drivers could be installed without signing (by simply disabling certificate authentication – a temporary solution suitable for development testing purposes), and the signing procedure was easier. For the later Windows versions, especially for Windows 10, a significant number of existing ways to build, sign (or disable signature verification) and install drivers do not provide a solution. However, such work can be performed on a system loaded in "test mode". To boot the system in test mode, you can run a command on the command line running as an administrator and restart the system:

```
bcdedit /set testsigning on.
```

Installation and signing of the driver is done in the test mode of the system. The signature can be done using the utility `SignTool.exe` by the `Sign` command.

This driver is called in the implemented

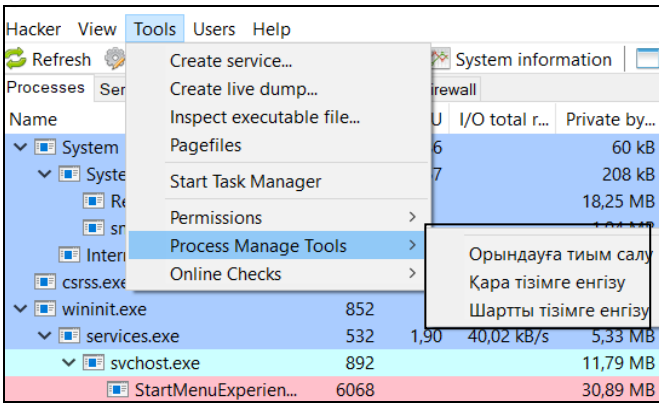

Fig. 2. Menu with plugin functionality

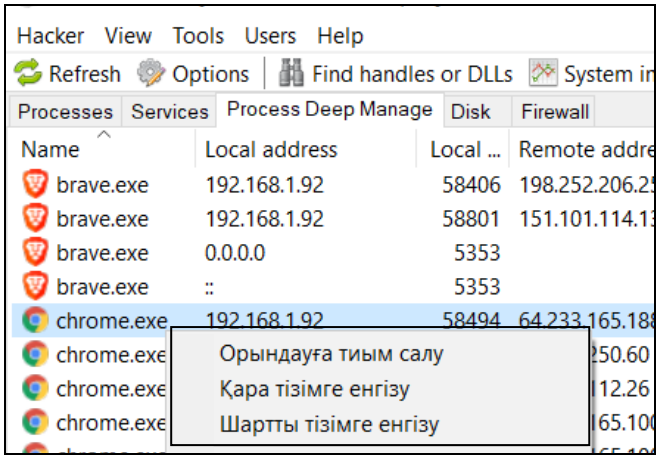

Fig. 3. Calling the plugin functionality from the context menu

```
#define PROCESSDEEPMANAGE_PLUGIN_NAME L"ProcessHacker.ProcessDeepManage"
…

VOID NTAPI ProcessDeepManageMenuInitializingCallback(
    _In_opt_ PVOID Parameter,
    _In_opt_ PVOID Context
)
…

PhInsertEMenuItem(menuInfo->Menu, forbidMenu = PhPluginCreateEMenuItem(PluginInstance, 0,
    PROCESSDEEPMANAGE _ACTION_FORBID, L"&Орындауға тиым салу", ProcessDeepManageItem), 0);
PhInsertEMenuItem(menuInfo->Menu, AddMenu = PhPluginCreateEMenuItem(PluginInstance, 0,
    PROCESSDEEPMANAGE _ACTION_Add, L"&Қара тізімге енгізу", ProcessDeepManageItem), 1);
PhInsertEMenuItem(menuInfo->Menu, AddToListMenu = PhPluginCreateEMenuItem(PluginInstance, 0,
    PROCESSDEEPMANAGE_ACTION_ AddToListMenu, L"&Шартты тізімге енгізу", ProcessDeepManageItem), 2);
PhInsertEMenuItem(menuInfo->Menu, PhCreateEMenuSeparator(), 3);
```

Fig. 1. Embedding the plugin in the process Hacker interface

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

ProcessDeepManage plugin. The correspondence of the menu items is also indicated in Table I.

Starting a process involves file operations. The driver implements the callback function of the filter driver, which prevents the necessary file operations. This function returns the `FLT_PREOP_COMPLETE` value, and setting of the `STATUS_ACCESS_DENIED` value of the `Data->IoStatus.Status` is required. A fragment of the callback function is shown in Fig. 4.

In order to stop the further chain call, the value return is applied `FLT_PREOP_SUCCESS_NO_CALLBACK`, then the filter driver manager does not call the subsequent callback functions.

```
if (Data->Iopb->MajorFunction == IRP_MJ_CREATE)
{
    if ((Data->Iopb->Parameters.Create.SecurityContext->DesiredAccess &
        (FILE_WRITE_DATA | // 0x0002
            FILE_WRITE_ATTRIBUTES | // 0x0100
            FILE_WRITE_EA | // 0x0010
            FILE_APPEND_DATA | // 0x0004
            DELETE | // 0x00010000
            WRITE_DAC | // 0x00040000
            WRITE_OWNER // 0x00080000
            )))
    {
        Data->Iopb->Parameters.Create.FileAttributes |= FILE_ATTRIBUTE_
        Data->IoStatus.Information = 0;
        Data->IoStatus.Status = STATUS_ACCESS_DENIED;
        return FLT_PREOP_COMPLETE;
```

Fig. 4. Fragment of the driver's callback function that controls the start of processes

IV. Results and practical prospects

Thus, the results are obtained in the work:

- Classification of operating system administration tasks.
- Comparative analysis of open source utilities and built-in administration tools. The utilities Process Hacker, Process Explorer, built-in task manager, as well as others are considered according to such criteria as functionality, the complexity of use, portability, cross-platform, customizability, open-source
- Development and the implementation of the plugin in the Process Hacker utility based on the modification of the filter driver provided by the Microsoft EWDK developer kit. For one of the popular open source utilities Process Hacker, a plugin has been developed that has added such functionality to this utility as creating and managing your own list of prohibited and conditionally allowed programs in the system and automating the management of their launch.

The results of the analysis of the functionality of utilities for the administration of operating systems and computer networks will be useful in terms of selecting the optimal set of correctly working tools, as well as for identifying administrative tasks that require automation and the use of specialized programs. Classification and comparative analysis of the tasks and related administrative tools of modern Windows operating systems will be useful for operating system administrators.

The Process Hacker utility is a popular tool among system administrators and advanced Windows users. The developed plugin will add such functionality as managing system processes called by executable files with the ability to set a schedule for tasks. Modification, assembly and debugging of the driver, development, testing and deployment of the plugin were performed. Based on the developed driver in the plugin, implementing for example, such interesting features as prohibiting the display of the process and changing its description in the system application “Task Manager”, allowing to stop the process only after entering a password is possible. The development is directly applicable and facilitates certain tasks of administration of Windows 7 SP1, Windows 8, Windows 10.

References

[1] M. Russinovich, A. Margosis. Troubleshooting with the Windows Sysinternals Tools. 2nd Edition. – Microsoft Press, 2016.
[2] B. Knittel, P. McFedries. Windows 10 In Depth. 2nd Ed. – Que Publishing, ISBN-10: 0789759772, 2018.
[3] A. M. Kenin. A practical guide for a system administrator. 2nd ed., reprint. and additional-St. Petersburg: BHV-Petersburg, 2013.
[4] A. Ospanova, A. Zharkimbekova, K. Sagindykov, M. Kokkoz. Implementation and Commercialization of the Results of the “Multidisciplinary Mobile Computer Classroom Based on Raspberry Pi” Project/ iJET – Vol. 15, No. 13, 2020, P. 116-135.
[5] A. Ospanova, A. Zharkimbekova, L. Kussepova1, A. Tokkuliyeva1, M. Kokkoz. Cloud Service for Protecting Computer Networks of Enterprises Using Intelligent Hardware and Software Devices Based on Raspberry Pi Microcomputers, unpublished.
[6] B. Knittel. Windows 7 and Vista Guide to Scripting, Automation, and Command Line Tools. – Que Publishing, ISBN-10: 078973728, 2010.
[7] I. V. Korobko. The System administrator's Guide to Windows Programming. - St. Petersburg: BHV-Petersburg, 2009.
[8] P. K. Bhardwaj, D. Kleiman, B. Barber, K. Andreou. How to Cheat at Windows System Administration Using Command Line Scripts. – Syngress, 2006.
[9] (Process Hacker Source Code. Available: github.com/processhacker/processhacker/)
[10] (Process Hacker Extra Plugins Source Code. Available: github.com/processhacker/plugins-extra)
[11] P. B. Prasad. Operating systems and systems programming. – India: Scitech publications, 2015.
[12] Dh. Dhamdhere. Systems Programming. – McGraw Hill Education, 2011.
[13] (Process Hacker Forum. Available: forum.ru-board.com/topic.cgi?forum=5&topic=49036&start=3860)
[14] S. M. Palakollu. Practical System Programming with C – Pragmatic Example Applications in Linux and Unix-Based Operating Systems. – Apress, 2021. ISBN: 9781484263204, 9781484263211.
[15] O. Vaticone. C++ System Programming Cookbook: Practical recipes for Linux system-level programming using the latest C++ features. – Packt Publishing Ltd, 2020. ISBN: 9781838648756, 1838648755
[16] P. Yosifovich. Windows 10 System Programming, Part 1. – leanpub.com, 2019.
[17] J. Bremer. Intercepting System Calls on x86_64 Windows. Available: jbremer.org/intercepting-system-calls-on-x86_64-windows/#lowlevel
[18] (Book Review: Windows Kernel Programming and Creating Drivers of Select Exercises. Available: truneski.github.io/post/2020/04/03/book-review-windows-kernel-programming-and-creating-drivers-of-select-exercises/)
[19] A. Bassov. Hooking the native API and controlling process creation on a system-wide basis. Available: codeproject.com/Articles/11985/Hooking-the-native-API-and-controlling-process-cre
[20] T. Opferman. Driver Development Part 1: Introduction to Drivers. Available: codeproject.com/Articles/9504/Driver-Development-Part-1-Introduction-to-Drivers

ISBN: 978-988-14049-2-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)