Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Системное программное обеспечение вычислительных машин (СПОВМ)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему:

«Сниффер клавиатуры»

Студент: гр.350501 Шешко В.Р.

Руководитель: Яночкин А.Л.

Минск 2015

Содержание

[Введение 3](#_Toc414805627)

[1.Обзор литературы 4](#_Toc414805628)

[2. Техническое задание 6](#_Toc414805629)

[Заключение 7](#_Toc414805630)

[Список литературы 8](#_Toc414805631)

# Введение

Общество, в котором живет современный человек, характеризуют как информационное. Компьютеры получают все более широкое применение – даже там, где, как казалось совсем недавно, их применить невозможно. Информационные технологии и их знание становится неотъемлемой частью жизни любого человека, и в основу применения этих технологий также легло применение компьютера.

Сегодня большая часть работы выполняется с использованием компьютера: электронных документов, файлов, программ и сайтов. Неудивительно, что хронометраж рабочего времени за компьютером так популярен сейчас. С его помощью можно проанализировать рабочий процесс и выявить резервы времени.

**Учет деятельности за компьютером** — дешевый и полностью объективный способ. Он идеально подходит для офисного персонала, который большую часть времени проводит на рабочем месте за компьютером.

Снифферы клавиатуры принадлежат к той группе программных продуктов, которые осуществляют контроль над деятельностью пользователя персонального компьютера. Программные продукты этого типа предназначаются исключительно для записи информации о нажатиях клавиш клавиатуры, в том числе и системных, в специализированный журнал регистрации (лог-файл), который впоследствии изучается человеком, установившим эту программу.

1. **Обзор литературы**

В этом разделе будут рассмотрены основные теоретические сведения, необходимые для создания программы «Сниффер клавиатуры».

* 1. Спецификация клавиатурных снифферов

Клавиатурный сниффер — это программное обеспечение или аппаратное устройство, регистрирующее нажатия клавиш на клавиатуре компьютера.

Клавиатурные снифферы подразделяются на аппаратные и программные. Первые представляют собой небольшие устройства, которые могут быть закреплены на клавиатуре, проводе или в системном блоке компьютера. Вторые — это специально написанные программы, предназначенные для отслеживания нажатий клавиш на клавиатуре и ведения журнала нажатых клавиш. Принципиальная идея сниффера состоит в том, чтобы внедриться между любыми двумя звеньями в цепи прохождения сигнала от нажатия пользователем клавиш на клавиатуре до появления символов на экране.

 Наиболее популярные технические подходы к построению программных клавиатурных снифферов:

* системная ловушка на сообщения о нажатии клавиш клавиатуры (устанавливается с помощью функции WinAPI SetWindowsHook, для того чтобы перехватить сообщения, посылаемые оконной процедуре.
* циклический опрос клавиатуры (с помощью функции WinAPI Get(Async)KeyState, GetKeyboardState).
* драйвер-фильтр стека клавиатурных драйверов ОС Windows.

Кейлоггеры могут внедряться в любом месте последовательности обработки, перехватывая данные о нажатых клавишах, передаваемые одной подсистемой обработки следующей подсистеме в цепочке обработчиков.

Установка ловушки для клавиатурных сообщений

Это самый распространенный метод реализации клавиатурных снифферов. Посредством вызова функции SetWindowsHookEx сниффер устанавливает глобальную ловушку на клавиатурные события для всех потоков в системе. Фильтрующая функция ловушки в этом случае располагается в отдельной динамической библиотеке, которая внедряется во все процессы системы, занимающиеся обработкой сообщений. При выборке из очереди сообщений любого потока клавиатурного сообщения система вызовет установленную фильтрующую функцию.

К достоинствам этого метода перехвата относится простота и гарантированный перехват всех нажатий клавиатуры, из недостатков можно отметить необходимость наличия отдельного файла динамической библиотеки и относительную простоту обнаружения по причине внедрения во все системные процессы.

Использование циклического опроса состояния клавиатуры

Состояние всех клавиш с небольшим интервалом опрашивается с помощью функций GetAsynсKeyState или GetKeyState. Данные функции возвращают массивы асинхронного или синхронного состояния клавиш; анализируя их, можно понять, какие клавиши были нажаты или отпущены после последнего опроса.

Достоинства данного метода — предельная простота реализации, отсутствие дополнительного модуля недостатки — отсутствие гарантии обнаружения всех нажатий, могут быть пропуски; легко обнаруживается мониторингом процессов, опрашивающих клавиатуру с высокой частотой.

Использование драйвер-фильтра драйвера класса клавиатуры Kbdclass

Документированный в DDK способ перехвата. Снифферы, построенные на основе этого метода, перехватывают запросы к клавиатуре посредством установки фильтра поверх устройства «\Device\KeyboardClass0», созданного драйвером Kbdclass. Фильтруются только запросы типа IRP\_MJ\_READ, поскольку именно они позволяют получить коды нажатых и отпущенных клавиш.

Достоинства — гарантированный перехват всех нажатий, невозможность обнаружения без использования драйвера. Недостатки — необходимость установки собственного драйвера.

При выборе средств для разработки приложения необходимо учесть множество различных аспектов, наиболее важным из которых является язык программирования, так как он в значительной степени определяет другие доступные средства. Для реализации клавиатурного сниффера наиболее эффективным будет использование языков программирования C и C++. Они предоставляют все средства для создания приложения для операционных систем семейства Windows X.

1. **Техническое задание**

1.1 Общие сведения.

Название: «Сниффер клавиатуры».

«Сниффер клавиатуры» − программа для слежения за активностью пользователя.

1.2 Назначения и цели программного средства.

Назначения: программа предназначена для отслеживания нажатия клавиш на клавиатуре, а также отслеживания изменения активного окна.

Цели: повышение производительности сотрудников на работе, при помощи контроля их активности. Родительский контроль.

1.3 Требования к программному средству

Данная программа должна вести мониторинг клавиатуры, отслеживать активные окна, активные вкладки в браузерах. Программа автоматически запускается при загрузке компьютера и работает незаметно для пользователя. Программа не должна отображаться в Диспетчере задач.

1. **Системное проектирование**

Данный проект включает в себя непосредственно два модуля: модуль для установки и инжекции динамической библиотеки в explorer.exe и сама динамическая библиотека. Динамическая библиотека содержит в себе сам обработчик событий клавиатуры и модуль работы с log файлами.

1. **Функциональное проектирование**

Данный раздел включает в себя описание основных методов программы и их листинги.

HHOOK **SetWindowsHookEx**( int idHook, HOOKPROC lpfn, HINSTANCE hMod, DWORD dwThreadId) - устанавливает определяемую программой процедуру фильтра (hook) в цепочку фильтров (hook).

idHook [in] Определяет тип устанавливаемой процедуры фильтра (hook). В курсовом проекте устанавливается процедура фильтра (WH\_KEYBOARD\_LL), которая осуществляет текущий контроль за низкоуровневыми событиями ввода данных с клавиатуры.

Lpfn [in] Указатель на процедуру фильтра (hook). Если параметр dwThreadId равняется нулю или устанавливает идентификатор потока, созданного другим процессом, параметр lpfn должен указывать на процедуру фильтра (hook) в динамически подключаемой библиотеке (DLL). В противном случае, lpfn может указывать на процедуру фильтра (hook) в коде, связанном с текущим процессом.

hMod [in] Дескриптор DLL, содержащий процедуры фильтра (hook), на которую указывает параметр lpfn. параметр hMod должен быть установлен в ПУСТО (NULL), если параметр dwThreadId устанавливает поток, созданный текущим процессом, и если процедура фильтра (hook) находится внутри кода, связанного с текущим процессом.

dwThreadId [in] Устанавливает идентификатор потока с которым, процедура фильтра (hook) должна быть связана. Если этот параметр равняется нулю, процедура фильтра (hook) связывается со всеми существующими потоками, запущенными на том же самом рабочем столе, что и вызывающий поток.

LRESULT CALLBACK LowLevelKeyboardProc(int nCode, WPARAM wParam, LPARAM lParam) - система вызывает эту функцию каждый раз, когда собирается вставить новое событие ввода с клавиатуры в очередь ввода данных потока. Ввод с клавиатуры может исходить от локального драйвера клавиатуры или от вызовов функции keybd\_event.

wParam [in] Устанавливает идентификатор сообщения клавиатуры.

lParam [in] Указатель на структуру KBDLLHOOKSTRUCT.

Листинг функции LowLevelKeyboardProc

LRESULT CALLBACK LowLevelKeyboardHook(int nCode, WPARAM wParam, LPARAM lParam)

{

GetKeyState(NULL);

if (nCode == HC\_ACTION && (wParam == WM\_KEYDOWN || wParam == WM\_SYSKEYDOWN))

{

DWORD written;

KBDLLHOOKSTRUCT \*ks = (KBDLLHOOKSTRUCT\*) lParam;

SYSTEMTIME st;

CHAR moduleName[MAX\_MODULE\_NAME];

DWORD winID = NULL;

HWND activeWindow = GetForegroundWindow();

GetWindowThreadProcessId(activeWindow, &winID);

GetWindowText(activeWindow, moduleName, MAX\_MODULE\_NAME);

if (winID != prevWinID || strcmp(moduleName, prevModuleName))

{

prevWinID = winID;

strcpy(prevModuleName, moduleName);

sprintf(moduleName + strlen(moduleName), " : %d", winID);

strcat(moduleName, "\r\n");

WriteFile(file, moduleName, strlen(moduleName), &written, NULL);

}

GetLocalTime(&st);

GetTimeFormat(LOCALE\_SYSTEM\_DEFAULT, 0, &st, NULL, g\_buf, MAX\_BUFSIZE);

strcat(g\_buf, " ");

char \* buf = g\_buf + strlen(g\_buf);

if (getSysKey(ks->vkCode))

{

WriteFile(file, g\_buf, strlen(g\_buf), &written, NULL);

}

else if (getCharKey(ks->vkCode, ks->scanCode, (LPWORD) buf, winID))

{

strcpy(buf + 1, "\r\n");

WriteFile(file, g\_buf, strlen(g\_buf), &written, NULL);

}

else

{

strcat(g\_buf, "[unknown]\r\n");

WriteFile(file, g\_buf, strlen(g\_buf), &written, NULL);

}

}

return CallNextHookEx(hook, nCode, wParam, lParam);

}

Структура KBDLLHOOKSTRUCT содержит информацию о низкоуровневом событии ввода данных с клавиатуры.

typedef struct {

DWORD vkCode;

DWORD scanCode;

DWORD flags;

DWORD time;

ULONG\_PTR dwExtraInfo;

} KBDLLHOOKSTRUCT, \*PKBDLLHOOKSTRUCT;

vkCode устанавливает код виртуальной клавиши. Код должен быть значением в диапазоне 1 - 254.

scanCode устанавливает аппаратный код опроса клавиатуры (скэн-код) клавиши.

flags устанавливает флажок дополнительной клавиши, флажок события ввода, контекстный код и флажок переходного состояния.

time устанавливает отметку времени для этого сообщения.

dwExtraInfo устанавливает дополнительную информацию, связанную с сообщением.

**Заключение**

# Список литературы