VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

SYSTÉM PRO MONITOROVÁNÍ ČINNOSTI UŽIVATELE

SYSTEM FOR MONITORING USER'S ACTIVITY

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR
ALIAKSANDR YEFTSIFEYEU

VEDOUCÍ PRÁCE
SUPERVISOR
ING. MICHAL POLÍVKA

BRNO 2011

Anotace

Bakalářská práce popisuje možnosti programů pro záchyt stisků kláves (keyloggerů) různého typu. Jsou-li tyto program nasazeny ilegálně, mohou být příčinou finančních a jiných ztrát. Obrana proti tomuto druhu monitorovacího softwaru je často složitá. Pro tvorbu softwaru sloužícího k obraně před tímto typem maligních programů je třeba znát principy a algoritmy, se kterými monitorovací software pracuje.

V rámci bakalářské práci byla podrobně rozebráná problematika vstupu/výstupu z klávesnice počínaje hardwarem a konče softwarovou aplikační vrstvou. Popsány jsou algoritmy záchytu kláves, základžené na systémových voláních operačního systému MS Windows, jejich vztah mezi zařízením a softwarem. Cílem bylo nalézt nejvýkonnější, resp. nejvhodnější algoritmus (ale také v současnosti nejpoužívanější), pro vytvoření softwaru pro sledování stisků kláves. Výsledkem je popis běžně používaných algoritmů. Byla zvolena jednoduchá, ale efektivní metoda, pro napsání vlastního programu. Projekt se zabývá problematikou záchytu stisků kláves z pohledu legálního i nelegálního sledování činnosti uživatele.

Protože bylo jedním z cílů projektu odeslání zachycených stisků kláves na vzdálený server, byl proveden rozbor problematiky přenosu dat na vzdálený uzel přes síťové rozhraní, za pomoci různých síťových protokolů v Ethernetu. Rámcově byly popsány protokoly TCP a UDP, a jejich role v přenosech dat po síti. Práce také zahrnuje popis protokolů vyšších vrstev síťového referenčního modelu OSI. Z různých hledisek byly diskutovány protokoly TFTP, FTP, SSL, SSH – jejich výhody, nevýhody a jejich využitelnost k přenosu zachycených stisků kláves. Analýza problematiky protokolů 5. a 6. vrstvy RM OSI vedla k volbě protokolu TFTP, jako protokolu nejvhodnějšího pro přenos malého objemu dat mezi monitorovacím programem a serverem. Protokol TFTP vyniká jednoduchostí a snadnou implementovatelností. Obvykle se používá jako servisní protokol pro přenosy konfigurací, logů a firmwarů v lokálních sítích.

Mezi nevýhody protokolu patří neexistence zabezpečení přenášených dat jak z hlediska možnosti vzniku chyb při jejich přenosu, tak z hlediska možnosti jejich odposlechu. Problematická je také bezpečnost dat na TFTP serveru Protokol nepočítá s autorizací a autentizací klienta. Zmíněné nevýhody jsou ale implementačními výhodami – implementace protokolu do vlastního programu je jednoduchá.

Nevýhody TFTP jsou vzhledem k použité aplikaci zanedbatelné. V případě nasazení při legálním odposlechu by byla data pravděpodobně odesílána pouze v rámci lokální sítě, jejich zabezpečení by mohl obstarat server zajišťující agregaci zachycených stisků kláves ze všech klientů. V případě nelegálního odposlechu je prioritou útočníka nenápadnost, v takovém případě by se pravděpodobně více hodil např. protokol SMTP nebo HTTP, resp. HTTPS.

Klientská část protokolu TFTP byla implementována do vlastního programu pojmenovaného Keylog. Program Keylog je hlavním produktem bakalářské práce, Keylog zachycuje všechny stisknuté klávesy, ať už se jedná o přihlašovací údaje k elektronickému bankovnictví v internetovém prohlížeči, heslo do firemního

informačního systému nebo dopis zákazníkovi. Zachycené klávesy ukládá do souboru a v určených časových intervalech je odesílá na server.

Keylog umožňuje analyzovat problematiku monitorování stisku kláves na klávesnici z pohledu detekce na pracovní stanici, i při přenosu zachycených dat. Na programu Keylog je ukázána problematika systémového útoku a obrana proti němu pomocí standardních nástrojů obsažených přímo v OS Windows – např. firewall, ochranné mechanismy proti modifikaci ovladačů klávesnice resp. systémových souborů, detekce automatického spouštění maligního softwaru atp. Podrobně bylo zkoumáno různé nastavení firewallu a navržena jeho bezpečná konfigurace.

Program Keylog, vytvořený v rámci bakalářské práce, slouží pouze jako modelový příklad a jeho nasazení při legálním monitoringu by vyžadovalo jeho úpravu. Program má prostý vzhled a jednoduché ovládání. Při práci s programem se zachycené klávesy ukládají do textového pole, resp. souboru, pro jejich pozdější analýzu a případné odesílání dat na vzdálený server. Odesílání dat je periodické (každých 5 minut je odeslán soubor s nově stisknutými klávesami). Celý soubor není odesílán, protože v takovém případě by objem přenesených dat neustále narůstal. Při případném praktickém nasazení by mohl být upraven časovač odesílání tak, aby odesílání probíhalo v pseudonáhodných intervalech tak, aby nebylo možné jednoduše detekovat periodické odesílání. Dále by mohlo být implementováno šifrování, aby byla detekce ztížena pro systémy IDS/IPS. Program byl testován s TFTP serverem od firmy SolarWinds, ale vzhledem k tomu, že protokol byl standardně implementován, neměl by vzniknout problém při použití programu s jakýmkoli jiným serverem. Program dále vypisuje statistiky odesílání dat – datum, čas odeslání atp.

V textové části projektu jsou podrobně popsány současné nejrozšířenejší legální i nelegální monitorovací systémy. Tyto poznatky byly zohledněny i při návrhu vlastního řešení. Textová část dále rozebírá možnosti obrany proti většině popsaných monitorovacích programů – proti vnitřním a vnějším útokům.

Při nasazení jakéhokoli legálního systému pro monitorování uživatelů, je třeba důsledně dbát na bezpečnost použitého programu. V případě, že by se podařilo útočníkovi převzít kontrolu nad tímto softwarem, mohlo by to mít pro provozovatele fatální důsledky. Legální monitorovací software se často používá ve státní správě a dalších společnostech ke sledování správného využívání výpočetní techniky zaměstnanci. V těchto společnostech ale ne vždy patří bezpečnost IT k hlavním prioritám. Obrana před převzetím kontroly nad zvoleným monitorovacím softwarem však nezáleží vždy přímo na použitém softwaru, často záleží také na konfiguraci operačního systému a dalším instalovaném softwaru. Bezpečnost programu Keylog byla testována – při správném nastavení Windows, je program neškodný.

Vytvořený program Keylog má jediný funkční nedostatek – je vytvořen bez využití vláknového programování, to znamená, že pracuje lineárně a nevykonává několik funkci naráz. To je také možným vylepšením vytvořeného programu. Dalším možným vylepšením by mohly být – náhrada použitého přenosového protokolu a doplnění informačního výpisu v průběhu práce programu.

Klíčová slova

Monitorování, spyware, uživatelský program, zachycení stisku kláves, protokol, siťový přenos, protokol TFTP, vzdálený server.

Аннотация

В рамках курсовой работы была создана программа Keylog, которая является модельным примером для характеристики функциональности программы с целью перехвата нажатых клавиш, а также последующей отправки файла, который содержит записанные данные на удалённый узел посредствам сетевых протоколов. Проект позволяет подробнее описать проблематику перехвата нажатых клавиш с точки зрения легального и нелегального мониторинга действий пользователя. Изучена техника отправления данных по заданному, удалённому IP адресу. Созданная программа служит для описания возможностей используемых функций и алгоритмов подобных программ. Подробно были изучены и описаны современные распространённые легальные и нелегальные мониторинговые системы, полученные результаты были использованы для создания собственного решения. Созданная программа позволяет сохранить полученные данные в текстовый файл и далее с ним работать. Отправлять в постоянном временном интервале записанные символы на удалённый узел. Текстовая часть подробнее разбирает данную проблематику, описывает отдельные алгоритмы, как процесса перехвата, так и отправки данных. Далее описывает возможности защиты от большинства описанных вредоносных программ, а также безопасность сетевой передачи.

Ключевые слова

Клавиатурный шпион, мониторинг, перехват, клавиши, программа, кейлоггер, функции перехвата, пользователь, протокол, удалённый узел, передача.

Abstract

As a part of the semester work, the program Keylog is made as a model for the characteristic features of the program to intercept keystrokes. The project allows a user to describe in details the problems of capturing keystrokes from the standpoint of legal and illegal monitoring of user's actions. This program describes the capabilities of the functions and algorithms of these programs. Details have been studied and described by modern common legal and illegal monitoring systems, the results were used to create their own solutions. The program allows a user to save the data in a form of a text file and continue working with it. The text describes in details this perspective, the individual algorithms, and furthermore the ability to protect against the most malicious programs.

Key words

Spyware, monitoring, hooking, keys, program, key logger, hook function, user.

YEFTSIFEYEU, A. *Systém pro monitorování činnosti uživatele.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. 38 stran, 2 přílohy. Vedoucí bakalářské práce Ing. Michal Polívka.

Prohlášení

Prohlašuji, že svoji bakalářskou práci na téma “Systém pro monitorování činnosti uživatele“ jsem vypracoval samostatně pod vedením vedoucího semestrální práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 1 1 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne . .....

podpis autora

Poděkování

Děkuji svému vedoucímu bakalářské práce Ing. Michalu Polívkovi za odbornou a pedagogickou pomoc a také za cenné rady při zpracování bakalářské práce.

V Brně dne…… ……

Aliaksandr Yeftsifeyeu

Оглавление

Список используемых иллюстраций

Список используемых таблиц

1. Введение в понятие клавиатурный шпион

Безопасность информации является основным приоритетом в любой деятельности. Прежде всего, это касается больших и малых организаций. Развитие цифровой сферы позволило передавать большие объёмы информации между пользователями быстро и легко вне зависимости от их расстояния. Это привело к возникновению людей, которые любыми возможными способами пытаются завладеть интересующей их информацией. Как правило, в этом случае достаточно получить удалённый доступ к банку данных. Но гораздо проще воспользоваться уже готовой программой, которая сама выполнит все необходимые для этого действия. Под особое внимание попадают именно коммерческие и военные объекты, которые, содержат наиболее важные сведенья, но, как правило, хорошо защищены. Пользователь же, сам отвечает за свой персональный компьютер, а также информацию, которая на нём содержится. Основной целью в данном случае являются пароли и прочие регистрационные данные. Сегодня, когда интернет получил широкое распространение по всему миру, наиболее незащищённой группой являются именно пользователи, ведь зачастую троянские программы достаточно сложно обнаружить, а методы защиты не всегда эффективны, злоумышленники придумывают всё новые способы получения информации у своих жертв. Одними из таких достаточно сложно обнаруживаемых программ являются клавиатурные шпионы или кейлоггеры[1], которые не распознаёт антивирусная защита или распознаёт не всегда.

Клавиатурный шпион это программа, способная скрытно фиксировать действия, а также вводимые данные пользователя и сохранять их в нужном формате, а впоследствии и передавать через службы связи злоумышленнику. Сам же по себе, клавиатурный шпион не является вирусной программой, он не копирует себя, как это делают другие вредоносные программы, но по своей опасности не уступает более вредоносным, ведь зачастую именно информация, несёт наиболее важную ценность для человека. В свою очередь, современные клавиатурные шпионы способны не только к записи нажатых клавиш пользователя, но также к сканированию его действий, “фотографирование” рабочего стола и записи об открываемых сайтах, это далеко не полный список их функций. Клавиатурные шпионы способны отследить открытые приложения, впоследствии, злоумышленник получает всю записанную информацию посредствам почтовых служб или других коммуникационных протоколов.

Для того, чтобы отследить исходящий трафик вполне достаточно обычного сниффера, но как разобраться какие именно пакеты представляют угрозу? Прежде всего, необходимо знать какие протоколы позволяют

[1] От английского keylogger

передавать локальные данные на удалённый узел. А также, какие для этого используются дополнительные средства, такие например как кодирование информации.

Но что если в целях именно безопасности используется троянская программа - клавиатурный шпион. Мониторинг своего же клавиатурного ввода в первую очередь может облегчить задачу запоминания ненужной информации, если таковой много, а также к последующему анализу введённых данных. Простота создания и его эксплуатация являются не последним положительным качеством этой программы.

2. Виды и описание клавиатурного ввода-вывода

Существует много разных типов клавиатурных шпионов, условно их можно поделить по типу скрытности и отслеживанию действий на компьютере пользователя. В зависимости от того, какой метод слежения выбран, можно разделить на следующие:

1. Слежение за клавиатурным вводом посредствам ловушек (hooks).
2. Слежение за клавиатурным вводом на основе циклического опроса клавиатуры.
3. Отслеживание ввода с помощью перехвата API функций.
4. Отслеживание ввода, который основан на специальных драйверах.

Так же, клавиатурные шпионы условно можно разделить по методу отправления зарегистрированных логов посредствам сетевых служб на:

1. Почтовые (посредствам почтового протокола).
2. Иные (в интернете или локальной сети посредствам протоколов FTP, HTTP и т.п.).

Современные клавиатурные шпионы, способны сочетать в себе несколько методов одновременно, для большего удобства и надёжности передачи [1], [3].

Перед тем как описывать каждый из этих типов стоит рассмотреть посредствам чего и как осуществляется ввод и вывод символов на экран. Предложенная аппаратная модель показана на рис. 1 [2].

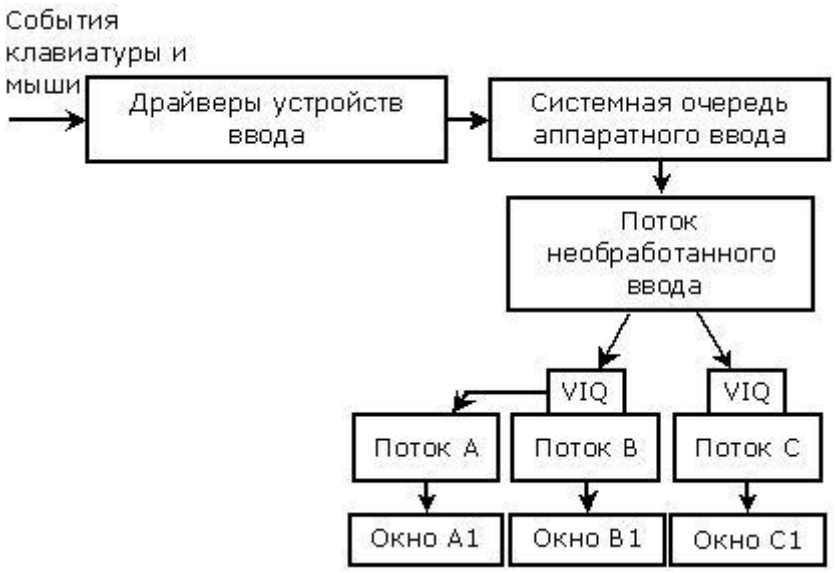

Рис. 1: Аппаратная модель из книги Д.Рихтера "Windows для профессионалов".

При нажатии на клавишу, возникает событие ввода и при отпускании событие вывода. Эти действия, обрабатываются драйвером устройства, который перемещает события в системную очередь аппаратного ввода SHIQ.

Далее из системной очереди ввода в поток необработанного ввода RIT. В обычном состоянии поток RIT бездействует и ожидает приходящего события с SHIQ, который в свою очередь, ожидает пользовательского события ввода. Данные потоки являются основой аппаратного ввода. После получения такого события RIT, преобразовывает полученные события из аппаратного ввода в сообщения. После этого, сообщения перемещаются в виртуальный ввод одного из трёх потоков. В который из этих потоков поместить данное сообщение решает RIT. Потоки A,B,C представляют собой активные потоки, т.е. потоки, с которыми пользователь работает в данный момент, посредствам окон A1, B1, C1. Таким образом, события ввода с клавиатуры могут быть отправлены одному потоку. На рис. 1 показан поток A, который не имеет очереди виртуального ввода. В этом случае потоки A и B используют общий виртуальный ввод. Каждый поток, содержит структуру THREADINFO. Данная структура используется для обмана потока, который будет считать, что его выполнение происходит в среде, которая принадлежит данному потоку. Другими словами THREADINFO это структура сопоставимая с потоком, которая выполняет функции определения очереди синхронных и асинхронных сообщений потока, а также, очередь виртуального ввода VIQ, очередь ответных сообщений (Reply-Message Queue), флаги пробуждений и прочее. В данной структуре также содержатся сведенья о состоянии ввода, различные для различных устройств [1], [2].

Конкретно для клавиатуры используются следующие данные:

- Какое из активных окон находится в фокусе клавиатуры
- Какое из окон активно в данный момент
- Какие клавиши были нажаты
- Состояние курсора ввода

Для мыши используются сведенья:

- В каком окне находится курсор мыши
- Какую форму имеет курсор мыши
- Является ли этот курсор видимым

Как уже известно, ввод с клавиатуры вызывает событие, которое обрабатывается драйвером и направляется в очередь аппаратного ввода, а оттуда в поток необработанного ввода. Далее RIT, преобразует событие в сообщение и ставит в очередь виртуального ввода одного из потоков, но не в окно, т.е. не важно, какому окну принадлежит поток. Если вызывается функция GetMessage, которая позволяет извлечь сообщение из потоковой очереди, то сообщение из потока изымается и отправляется конкретному окну, в котором находится клавиатурный фокус ввода. Если пользователь хочет получить сообщение в любом другом окне, то необходимо заранее настроить в потоке необработанного ввода, какому окну данное сообщение должно отправляться. То окно, которое уже больше не будет использоваться, гасит

курсор ввода, а окно, получающее впоследствии фокус, показывает курсор ввода. Следует также учитывать, что RIT единовременно отправляет сообщение только одному такому потоку. Также за обработку сочетаний клавиш таких, например как Alt+Tab и Alt+Ctrl+Delete отвечает поток необработанного ввода. [2]

3. Описание принципа работы клавиатурного шпиона по типу слежения

Как было сказано в главе 2, клавиатурные шпионы можно разделить на несколько групп по выбранному средству слежения за действиями пользователя на несколько типов. Теперь остановимся более подробно на каждом из них.

3.1. Слежение за клавиатурным вводом посредствам ловушек.

Построение программы клавиатурного шпиона на основе перехвата сообщений является классической. Суть состоит в том, чтобы с помощью вызова функции SetWindowsHookEx, которая описана в библиотеке user32.dll, перехватить обмен сообщениями между потоками. Также, при установки ловушки необходимо определить тип сообщений, для которых будет вызываться данная функция. Такими сообщениями будут являться WH_KEYBOARD и WH_MOUSE. Ловушка может быть установлена на один поток или на все сразу. Написание клавиатурного шпиона на основе ловушки является достаточно эффективным способом, так как многие из существующих антикейлоггеров не рассчитаны на поиск перехвата функций. Из недостатков необходимо выделить обязательное наличие библиотеки DLL, в которой должен содержаться код обработчика событий ловушки, при использовании которой происходит проецирование во все процессы GUI сразу после получения сообщения, что делает данный метод более обнаруживаемый [1].

3.2. Слежение за клавиатурным вводом на основе циклического опроса

Заключается метод в опросе состояния клавиш на событие ввода-вывода посредствам функций ввода с клавиатуры (Keyboard Input Functions) таких как: GetKeyboardState, GetAsyncKeyState, GetKeyState. Например, функция GetKeyboardState, при возвращении параметра массива, содержит одну из 256 возможных кодов виртуальной клавиши. Высший бит это 1, событие происходит, когда клавиша нажата, низший 0, когда клавиша отпущена. В случае с индикаторными клавишами наоборот, низшим битом считается 1 и показывает переключение клавиши, а высшим 0, когда клавиша не переключена. Таким образом, события обрабатываются в порядке нажато-отпущено [8].

3.3. Отслеживание ввода с помощью перехвата API функций

Заключается в подмене некоторого адреса в системе или кода в API функции для того, что бы при вызове этой функции управление передавалось не ей, а пользовательской функции. После этого, пользовательская функция вместо системной будет выполнять заранее заданные действия пользователем. Далее возможен дальнейший вызов системной функции или без вызова вообще.

Метод осуществляется подобным способом как с ловушками, написанием DLL файла с выполняющей действия функцией.

Данный метод достаточно сложен, поэтому широкого распространения в написании клавиатурных шпионов так и не получил. Однако является вполне функциональным и эффективным в силу того, что является достаточно скрытным и малозаметным шпионом, поэтому вполне способен применяться для написания клавиатурных шпионов [9].

3.4. Отслеживание ввода, который основан на специальных драйверах

Это один из наиболее эффективных клавиатурных шпионов, основанный на написании специальных драйверов. Установка возможна написанием специальных драйверов и последующей их замены или же использованием драйвера фильтра. Принцип работы драйвера фильтра схож с принципом перехвата функций, в которых содержится сообщение, только вместо сообщений используется непосредственно события ввода-вывода, которые обрабатывает драйвер после нажатия и отпускания клавиши. Реализация драйвера фильтра достаточно проста. Также возможна имплементация программы драйвера-шпиона в клавиатурный контроллер.

При написании полностью автономного драйвера и последующий его инсталляцией в систему требуется определиться с типом его установки в данную систему. Ведь зачастую это делает подобный драйвер легко обнаруживаемым. Драйвер может инсталлироваться со стандартного инсталлятора с последующей самоликвидацией инсталлятора либо же быть установленным посредствам INF файла. INF файл это стандартный файл OS Windows, который по своему формату схож с файлом INI, являясь по сути текстовым файлом и применяясь исключительно для информационных либо инсталляционных функций. Для написания любого драйвера используется файл INF. При этом, используя данный файл, появляется возможность создания самостоятельной утилиты установки. Теперь о том, как осуществляется его скрытность, ведь для того что бы такой файл инсталлировать, пользователь должен не догадываться о несущей им функции. Первая возможность осуществления этого, является подмена оригинального клавиатурного драйвера, драйвером собственной разработки. Раньше, сложность данного метода заключалась в использовании клавиатуры от разных разработчиков пользователем, что могло привести к конфликту. Но с появлением Windows XP, которая имеет встроенные стандартные драйвера для разных типов клавиатур и технологию самоидентификации оборудования посредствам технологии plug&play, эта проблема исчезла сама по себе. То есть теперь достаточно заменить INF файл, хранящийся в системной папке Windows на файл своей разработки.

В заключение, можно добавить, что данный тип слежения хоть и является эффективным и более простым в реализации, одновременно это делает его довольно легко обнаруживаемым [1], [5], [6], [7].

4. Реализация сетевой передачи данных

Передача данных через сетевые службы реализована с помощью двух основных протоколов работающих на транспортном уровне, а именно TCP и UDP. Сами по себе, эти протоколы являются средним, промежуточным звеном в транспортировке данных и выполняют лишь функцию обмена пакетов с приложениями более высокого уровня. Протоколы имплементированы в драйвер как функция. При необходимости передачи данных, происходит вызов соответствующей функции с указателем на буфер данных. Далее функция делит данные на сегменты (блоки данных) определённой величины и передаёт сегменты протоколу IP. На другом конце, при приёме, TCP/UDP собирает полученные данные в одно целое и отправляет далее приложению.

Процесс передачи и приёма данных довольно прост, но обладает рядом различных тонкостей, в первую очередь это зависит от выбора протокола, будь TCP или UDP. Самая главная особенность протокола TCP в отличие от UDP это установление соединения между узлами, а также последующее подтверждение данных при их приёме удалённым узлом. Протокол TCP создаёт пакеты ACK на противоположной стороне, которые в обязательном порядке отсылаются передающему данные узлу, тем самым контролируется передача и в случае не подтверждения посылка повторяется. Каждый пакет ACK передаёт последовательность подтверждаемых пакетов. Как правило, это может быть 1 или 0 соответственно для чётных или нечётных пакетов. Перед отправкой данных, TCP посылает запрос на удалённый узел с целью начать передачу, после этого, создаётся виртуальный канал, которому присваивается значение и начинается передача [12].

Рис. 2: Передача данных посредствам протокола TCP

Протокол UDP во многом схож с TCP, однако имеется довольно существенное отличие. При передачи каждый пакет получает адрес отправителя и нумеруется, тем самым данные передаются без предварительно установленного соединения, что делает UDP более простым. В таком случае гарантии доставки данных отсутствуют, тем не менее, предполагается низкая вероятность их утраты. Протокол UDP, также как и протокол TCP, работает с виртуальным каналом (порт) [14].

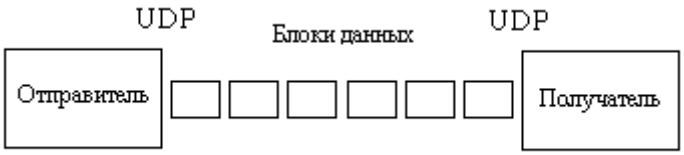

Рис. 3: Передача данных посредствам протокола UDP

4.1. Протоколы передачи данных в сети Ethernet

Так как протоколы TCP/UDP предоставляют лишь основную возможность для передачи данных, то есть имеют основные средства для начала сеанса передачи, они не позволяют обеспечить надёжную защиту передаваемым данным, а также, гарантировать их доставку. Для дополнительной надёжности передаваемого контента, например, для зашифровки (защиты) данных, а также передачи значительных объёмов разных данных, необходимо использовать расширенные для этого инструменты, например другие возможные стандарты сетевых протоколов. Для этих функций подходят более сложные протоколы прикладного и сеансового уровней, такие как FTP, TFTP, SSH, которые используют функции TCP/UDP.

Наиболее старший протокол передачи данных – FTP. Он позволяет подключаться к удалённым узлам, просматривать каталоги и загружать удалённые файлы. Передача данных происходит протоколом TCP через порт 20 и 21, последний из которых используется для ввода команд. Проблема данного протокола заключается в том, что он не является достаточно надёжным, то есть он не поддерживает шифрование данных. При подключении к удалённому серверу регистрационные данные передаются открытым текстом. В данном случае, при использовании сниффера, возможно перехватить пакет, несущий данные подключения. Однако проблема решается использованием протокола SSL, который создаёт безопасное соединение между сервером и клиентом. Данный протокол внедрён во все современные FTP сервера, такая связка очень часто именуется как FTPS протоколом, подчёркивая наличие интегрированного безопасного протокола SSL. Сам по себе, протокол SSL является довольно надёжным и устойчивым к атакам.

Протокол SSH используется для удалённого управления, а также, для передачи файлов. Является надёжным протоколом, позволяющим также сжимать передаваемые данные. По функциональности, его можно сравнить с протоколом Telnet, но позволяющим шифровать регистрационные данные, такие как логин и пароль.

Стандарт протокола SFTP является частью протокола SSH, позволяющим передавать данные, копировать и т.д.

Более простым протоколом для передачи данных является TFTP. Рекомендация протокола описана в приложении RFC 1350 [16]. Данный протокол используется в системном оборудовании, таком как свитчи, роутеры.

Выгода данного протокола заключается в его простоте, так как основная цель протокола это передача конфигурационных файлов или файлов настройки, а также служебных данных. TFTP протокол не поддерживает аутентификацию в отличие от FTP. Протокол TFTP основан на протоколе UDP, для более простой его реализации. Возможный недостаток данного протокола заключается в его беззащитности, фактически сервера построенные на данном протоколе имеют только фильтрование по входящим IP адресам [13].

4.1.1. Протокол TFTP и его свойства

Сам протокол был создан в 1983 году, для обслуживания сетевого оборудования и должен был служить загрузчиком для бездисковых рабочих станций. С развитием времени, протокол претерпел значительные изменения, так объёмы передаваемых данных, а также пропускная способность передачи увеличились, что привело к выходу более современных рекомендаций для данного протокола. Некоторые из них лишь слегка дополняют оригинал, а некоторые значительно его изменяют.

Стандарт протокола TFTP включает в себя разные виды служебных пакетов для разных целей. Всего определено 5 типов. Каждый из пакетов имеет определённую длину и несёт в себе отдельную функцию. Передача данных происходит в несколько этапов. Протокол использует пакеты-запросы для записи/чтения файла на сервере (WRQ/RRQ). Каждый такой пакет имеет определённую длину и свой номер в зависимости от того, для чего он предназначается (записи или чтения), а также несёт в себе информацию о данных, будь то название файла или способ передачи. В табл. 1 показано строение пакета-запроса.

После того, как сервер получил пакет с запросом на чтение или запись, он отправляет обратно пакет подтверждения, называемый ACK пакет с номером подтверждённого принятого блока данных, то есть для первого пакета это значение равно нулю. Далее начинается передача данных на сервер. Размер передаваемых данных определён в 512 байт, таким образом, данные передаются в блоках, при этом последний блок данных будет иметь размер менее 511 байт (516 байт с отводимыми на значение Opcode и номер отправляемого блока). Каждый принятый блок сервер подтверждает отправкой к клиенту ACK пакета с номером принятого блока, при этом клиент обязательно должен дождаться отправленного сервером подтверждающего пакета. Как только был получен последний блок - передача заканчивается. Такой способ отправки данных называется “stop-and-wait” и используется в наиболее простых протоколах.

5. Реализация и решение проекта в рамках курсовой работы

5.1. Создание сетевой передачи программой Keylog

Программа Keylog осуществляет передачу данных с помощью ввода IP адреса или доменного имени в текстовое поле под главной интерфейсной панелью программы. Алгоритм реализован с помощью функции Upload, которая запускается при нажатии кнопки “Start timer”. На удалённом узле, должен быть запущен TFTP сервер, который способен принять входящий запрос. После этого, значение записывается в переменную класса IpEndPoint, которая определяет конечную сетевую точку по IP адресу и порту. Протокол TFTP использует статический порт 69, который уже задан в программе и далее не требует ввода. При этом используется класс EndPoint, который представляет собой конечную точку для отправляемых данных. Существует возможность задания доменного имени сервера, который в последующем функцией Resolve будет переведён в IP адрес. Создание соединения осуществляет функция Socket, которая определяет тип и протокол передачи. На рис. 4 показан алгоритм создания сетевого соединения.

```
int Upload(String^ remotefile, String^ localfile, Modes
tftpMode)
     {
     int port = 69;
     int len = 0;
     int packetNr = 0;
     char t[128];
     time_t t_;

     <код был сокращён>

IPHostEntry^ iphostEntry = Dns::Resolve(Hostname->Text);
          IPEndPoint^ serverEndPoint = gcnew
          IPEndPoint(iphostEntry->AddressList[0],port);
          EndPoint ^ dataEP = (EndPoint^)serverEndPoint;
          Socket^ tftpSocket = gcnew Socket(serverEndPoint-
     >Address->AddressFamily,SocketType::Dgram,
     ProtocolType::Udp);

     <код был сокращён>
     }
```

Рис. 4: Инициализация данных и использование сетевых функций

Далее вызывается функция, которая описывает первый отправляемый запрос пакет (WRQ). Пакет состоит из нескольких частей, основные из которого Opcode и Filename.

Табл. 1: Структура запрос пакета

	2 bytes	string	1 byte	string	1 byte
WRQ / RRQ	01/02	Filename	0	Mode	0

Для запроса на запись, используется только один WRQ пакет. В программе Keylog отсутствует возможность принимать файлы, а значит и пакет RRQ. Как только пакет создан, создаётся сокетовое соединение и с помощью функции SendTo отправляется содержимое WRQ пакета с определёнными правилами UDP соединения на удалённый узел, записанный в переменной типа IPEndPoint. После этого, клиент ждёт ответа от сервера в течение 15 секунд. В этом промежутке, клиент должен принять отправленный сервером ACK пакет, который подтвердит принятый WRQ пакет номером нулевого блока. После подтверждения, можно начать цикл передачи данных на сервер. В случае возникновения исключения в процессе запроса ошибка будет отображена в текстовом поле richTextBox. Для создания пакета необходимо использовать несколько входящих параметров, как показано в табл. 1 тип и начало пакета определяется с помощью двух байтов, далее следует имя отправляемого файла, 1 байт для определения конца файла, далее тип передачи и последний байт определён для конца пакета. На рис. 5 показан программный код, которым создаётся запрос пакет.

```
char *Create_requestPacket(Opcodes opCode, String^
remotefileName, Modes tftpmode, int &ref_size)
{
String^ modeAscii = tftpmode.ToString()->ToLowerInvariant();
  char *ret;
  ref_size = modeAscii->Length + remotefileName->Length + 4;
  ret = new char[ref_size];
  sprintf (ret, "%c%c%s%c%d%c", 0, (byte)opCode,
remotefileName, 0, tftpmode, 0);
     return ret;
}
```

Рис. 5: Код генерирования запрос пакета

Цикл создания и отправки пакета данных построен на 3 этапах. Принятие подтверждающего пакета ACK и его оценка, генерирование и отправка пакета данных (Data packet) а также упаковка данных в блок 516 байт. Пакет данных несёт в себе код пакета, массив записанных символов и номер блока. Структура пакета показана в табл. 2:

Табл. 2: Структура пакета данных

	2 bytes	2 bytes	n bytes
DATA	03	Block #	Data

5.1.1. ОТПРАВКА ДАННЫХ НА СЕРВЕР

Как только сервер готов к передаче, возможно, запустить таймер программы Keylog нажатием кнопки “Start timer”, который через определённый промежуток времени будет передавать данные на заданный узел в текстовом поле. При этом, в главном текстовом поле будет отображаться справка о передачи файла, а именно дата и время отправки, а также удалённый узел с указанием IP адреса и порта, на который производится отправка. Перед отправкой, записанные знаки до текстового поля richTextBox сохранятся автоматически. Остановка сервиса производится нажатием кнопки Stop Timer во вкладке Internet на панели программы.

5.2. РЕАЛИЗАЦИЯ ПЕРЕХВАТЫВАНИЯ НАЖАТЫХ КЛАВИШ В ПРОГРАММЕ KEYLOG

Был выбран способ слежения за действиями пользователя посредствам опроса клавиатуры. Данный метод позволяет гибко настроить перехват нажатых клавиш с помощью функций GetKeyboardState, GetAsyncKeyState, GetKeyState, которые определены и описаны Microsoft. Каждая из них обладает разными свойствами, хотя в целом они очень схожи, все они принадлежат к Keyboard Input.

При написании программы Keylog, использовалась функция GetAsyncKeyState, но для начала, хотелось бы подробнее описать различие каждой из выше приведённых функций.

Функция GetKeyboardState. Дословно “Получить Состояние Клавиатуры”. Позволяет получить состояние всех клавиш посредствам массива в 256 виртуальных значений, соответствующие 256 клавишам клавиатуры. Для изменения состояния необходимо чтобы поток удалил сообщения из очереди сообщений, проводится с помощью функций PeekMessage, GetMessage. После того, как значение получено, копирует его в буфер данных, из которого по мере надобности данное значение может быть изъято. Если функция проведена успешно, то возвращает единицу, если нет – ноль.

Для достаточно эффективной работы с данной функцией необходимо, чтобы цикл опроса состояния клавиатуры был как можно больший, т.к. иначе, в случае, если пользователь печатает быстро, возможен пропуск знаков возвращающихся в массив. Так как GetKeyboardState возвращает состояние всех клавиш, а мне нужно отдельно каждой, эффективнее воспользоваться двумя другими.

GetKeyState (Получить Состояние Клавиши). Функция получает значение одной из виртуальных клавиш в зависимости от события, нажата ли клавиша, отжата или переключена. Как и GetKeyboardState, данная функция получает состояние клавиши после того как будет отобрано сообщение из очереди сообщений с помощью функций PeekMessage, GetMessage. Функция не показывает, какая клавиша нажата в данный момент, а лишь указывает, какая была нажата в предыдущем. Тоесть фактически была ли нажата клавиша

или нет. Для того, чтобы узнать, какая клавиша была нажата и какая нажата сейчас (начиная с последнего вызова функции) больше подойдёт GetAsyncKeyState.

GetAsyncKeyState (Получить Асинхронно Состояние Клавиши). Определяет, нажата ли клавиша во время вызова функции и была ли нажата с момента последнего вызова. Возвращает ноль в том случае, если фокус клавиатуры имеет окно в другом потоке. Именно эта функция отлично подходит для написания простого клавиатурного шпиона [11].

5.3. Структура программы и функции программы

Для успешной реализации и последующего написания клавиатурного шпиона, как и любой другой программы, необходимо, прежде всего, нарисовать блок-схему программы, которая покажет работу самого алгоритма. В приложениях на рис. 10, показана основная диаграмм-схема программы, которая реализует её основные действия. Программа начинается с входа из внешней среды в начало программы, что определено словом BEGIN. На этой стадии происходит загрузка программного процесса в системную память. После этого перед пользователем предстаёт программный интерфейс, который позволяет выполнить одно из предопределённых программистом действий, какое конкретно выбирает сам пользователь.

Так как программа рассчитана на работу с текстовым файлом, для запуска программы, необходимо либо создать необходимый файл с названием log.txt самостоятельно либо использовать кнопку Create file из соответствующего меню File. При этом созданный файл должен находиться в непосредственно той директории, где находится сама программа. При использовании кнопки автоматического создания файла “Create file”, программа автоматически создаст файл в нужной директории. При этом проверяется путь, в котором будет создан файл log.txt. В случае уже созданного файла, при нажатии кнопки “Create File” появится сообщение о созданном файле. Предположим, что файл был уже создан пользователем, тогда можно приступить к выполнению основной функции программы – перехвату нажатых клавиш. Для этого необходимо использовать кнопку RUN. После нажатия этой кнопки, программа свернётся в системный трей (расположен в правом нажнем углу на панели задач) и оповестит пользователя о том, что для последующего её извлечения необходимо использовать комбинацию клавиш ESC+F12 (см. рис. 6).

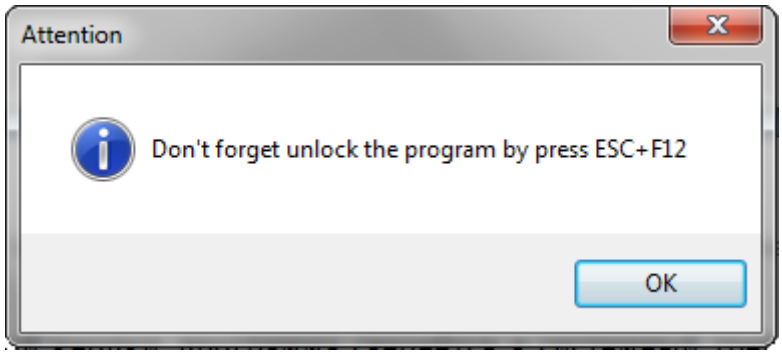

Рис. 6: Автоматическое оповещение

Оповещение происходит посредствам вызова функции MessageBox, которая задаёт параметры для окна оповещения. Главная же функция для этого действия называется hook, в которой написан цикл для перехвата отдельных клавиш с помощью GetAsyncKeyState.

Для создания постоянного цикла использовалась кнопка RUN, которая при сворачивании программы в системный трей возвращает значение циклической функции while - false. Это означает, что данная кнопка была нажата и пока программа свёрнута в трее возвращаемый параметр кнопки RUN является false. Далее программа находится в режиме готовности перехвата нажатых клавиш. Здесь я хотел бы привести пример из кода программы на рис. 7, в котором задействована функция GetAsyncKeyState:

```
if(GetAsyncKeyState(0x41))
          {
               richTextBox1->AppendText("a");
          }
          if(GetAsyncKeyState(0x42))
          {
               richTextBox1->AppendText("b");
          }
```

Рис. 7: Программный код с функцией GetAsyncKeyState

Программа осуществляется следующим образом. Текстовое поле richTextBox указывает на элемент, который реагирует на изменения в текстовом поле и с помощью функции AppendText, которая отвечает за добавление текста в элемент richTextBox1_TextChanged, добавляет перехваченные клавиши функцией GetAsyncKeyState. Сама по себе функция GetAsyncKeyState содержит адрес нажатой клавиши из таблицы ASCII и с помощью условия if проверяется, была ли нажата клавиша с определённым адресом указанным в функции GetAsyncKeyState. Если клавиша была нажата, то richTextBox1 добавляет символ в richTextBox1_TextChanged, какой конкретно символ добавится в текстовый элемент - определяется программистом. В конце цикла должна присутствовать функция Sleep. Она на время, заданное

в миллисекундах, усыпляет функцию GetAsyncKeyState, что позволяет задать интервал для последующего перехвата.

Если же были нажаты клавиши ESC+F12 (что также отслеживается с помощью GetAsyncKeyState) кнопка RUN возвращает значение true и цикл while – прекращается. В элементе richTextBox показываются значения нажатых клавиш. После этого, функция возвращается в первоначальное состояние и готова к дальнейшему использованию.

Сохранение данных в созданный файл log.txt пользователь производит самостоятельно. Функция сохранения реализована с помощью выбора в меню программы значения Save File. После того как было выбрано Save File, запустится функция ShowDialog, которая покажет окно выбора директории сохранения файла. Так же, возможно выбрать формат сохраняемого файла из двух: txt, rtf. После успешного сохранения файла возможно дальнейшее использование программы. Подробнее схема выполнения алгоритма Save File показана на рис. 11.

Последнее из возможного выбора в интерфейсе программы значение Exit завершает работу программы, но также позволяет сохранить полученный текст в результате работы клавиатурного шпиона. В случае выбора из окна Messagebox значения Yes выполняется функция saveFileDialog, которая запускает ShowDialog и далее пользователь может выбрать директорию сохранения файла. После сохранения результата производится автоматическое закрытие программы. Cancel позволяет отменить окно выбора и вернуться к интерфейсу программы. Если пользователь выбирает No, то программа закрывается без сохранения.

6. Методы защиты

6.1. Внутренняя защита от шпионского софта

Для успешного противодействия шпионским программам необходимо соблюдать все те же методы защиты, которые используются при обнаружении обычных вредоносных программ. Существует специальный софт для обнаружения клавиатурных шпионов, для его большей эффективности, надо помнить:

1. Использовать продукт известных производителей антишпионских программ
2. Производить частое обновление сигнатурных баз, желательно автоматически

Также при возможности необходимо проводить проверку системы на наличие шпионских программ. Необходимо помнить, что какой бы именитой не была программа антишпион нельзя быть полностью уверенным в том, что угрозы от кейлоггеров не существует.

Для максимальной защиты и наибольшей эффективности рекомендуется использовать нетрадиционные методы противодействия шпионским программам. В частности:

1. Использование одноразовых паролей с помощью генератора паролей
2. Использование виртуальной клавиатуры

При использовании одноразового пароля злоумышленнику будет достаточно трудно перехватить именно действующий пароль, обновление и отправка данных на адрес злоумышленника обычно ограничена. Однако такой метод может дать гарантию лишь на некоторый промежуток времени. Использовать можно различные генераторы паролей и брелки в их виде.

Использование виртуальной клавиатуры также может дать хорошую защиту на некоторое время. Виртуальная клавиатуры это программа, содержащая в себе все виртуальные клавиши обычной клавиатуры, которые необходимо нажимать мышкой. Однако для таких целей как гарантия защиты от перехвата данных, стоит использовать специальную клавиатурную программу, которая исключит возможность перехвата ввода-вывода из потока и передаче этих данных [3], [4].

6.1.1. Проактивная защита

Эффективной защитой от клавиатурного шпиона является, несомненно, проактивная защита, которая, однако, является инструментом опытного пользователя. Этот метод позволяет контролировать деятельность всех программ в зависимости от выполняемых ими действий в данный момент времени, а также проводить мониторинг системного реестра. Проактивная

защита – мощный инструмент, однако это требует определённых технических знаний со стороны пользователя и если таковых не имеется, то эффективность данного метода значительно уменьшается [10].

6.2. Сетевая угроза и защита от подобного софта

От сетевых атак, а также утечки данных с сетевого интерфейса, прежде всего, необходимо соответствующе настроить доступные средства Windows, такие например как Firewall, MS Windows Defender. Они предоставляют надёжную защиту от утечки нужной информации на отдалённые узлы, а также предохраняют компьютер от несанкционированного доступа. В частности решаются проблемы:

1. Доступ к внешним узлам, внутреннему контенту посредствам фильтрирования исходящего трафика и входящего трафика.
2. Фильтрация доступа от недостоверных служб
3. Уведомление о разного рода деятельности имеющихся программ и сетевых угроз извне.

В ряде случаев могут быть заблокированы нужные для работы порты разных сетевых служб, таких например как FTP, Telnet, TFTP, поэтому при возникающей ошибке в соединении прежде всего следует обратить внимание на эту службу.

Однако, существует также ряд проблем, которые Firewall решить не в состоянии, такие например как:

1. Неспособность защиты пользователя от загрузки вредоносных программ
2. Возникающие внутренние угрозы

Для решения данных проблем применяются внутренние антивирусные программы, такие как Microsoft Security Essentials или Windows Defender. В новой ОС Windows 7, Windows Defender был добавлен в антивирусную программу “Security Essentials” и являясь его постоянной частью, не требует отдельной установки.

Поэтому для исключения сбоев в операционной системе необходимо, вовремя закрывать неиспользуемые порты, а также, вовремя прекращать действия несанкционированного ПО[2] [15].

[2] ПО – Программное Обеспечение

7. Тестирование программы и её отладка

7.1. Настройка сервера

Для тестирования программы Keylog, использовался бесплатный и доступный для скачивания TFTP сервер от фирмы SolarWinds. После этого, необходимо соответствующим образом настроить систему безопасности Windows Firewall на сервере, а также, на компьютере клиента с программой Keylog. Следует учитывать операционную систему на которой будет устанавливаться сервер для настройки программного средства Firewall.

Инструкция пользователя для настройки сервера:

1) Скачать[3] по ниже приведённой ссылке TFTP сервер.
2) Запустить инсталляцию и установить его.
3) По желанию выбрать каталог, куда будут отправлены последующие тестовые логи, по умолчанию C:\TFTP-Root.
4) Запустить сервер с помощью кнопки Start во вкладке File->Configure->General->Start.
5) Дождаться запуска программы, до появления сообщения о старте.
6) Настроить MS Windows Firewall в зависимости от ОС[4].

Для MS Windows XP:

Control Panel->Windows Firewall->Exceptions->Add Port, в поле Name написать Keylog, порт 69, протокол UDP, далее OK.

Для MS Windows 7:

Control Panel->Windows Firewall->Allowed programs, создать исключение для программы Keylog.

7.2. Настройка программы Keylog

Программа была протестирована на связки ОС: Windows XP – Windows XP, Windows 7 – Windows XP, ошибок в течение передачи выявлено не было. На рис. 8 изображён интерфейс программы сразу после её запуска.

Инструкция пользователя для настройки программы Keylog:

1) Скопировать программу с CD на локальный диск.
2) Запустить и выбрать во вкладке File опцию Create new file, появится сообщение об успешном создании файла в директории программы.
3) В поле Hostname необходимо написать IP адрес сервера с установленной программой от фирмы SolarWinds.
4) Создать исключение для ОС Windows (см. п. 6)

[3] http://www.solarwinds.com/products/freetools/free_tftp_server.aspx
[4] ОС – Операционная Система

5) Запустить таймер нажатием кнопки “Start timer”, после этого каждые 15 секунд будет появляться сообщение об отправки файла log.txt на сервер, однако только в одном потоке, т.е. программа не должна быть свёрнута в системном трее.
6) Программу можно использовать нажатием кнопки “Run and minimize”.
7) По завершении работы с программой нажать комбинацию клавиш ESC+F12 для её разблокировки.
8) Дождаться отправки файла и проверить его наличие на сервере.
9) Остановить таймер можно во вкладке Internet опция “Stop timer”.

Если в течение передачи произойдёт исключение, то необходимо отключить защиту внутренней (приватной сети) в ОС Windows 7.

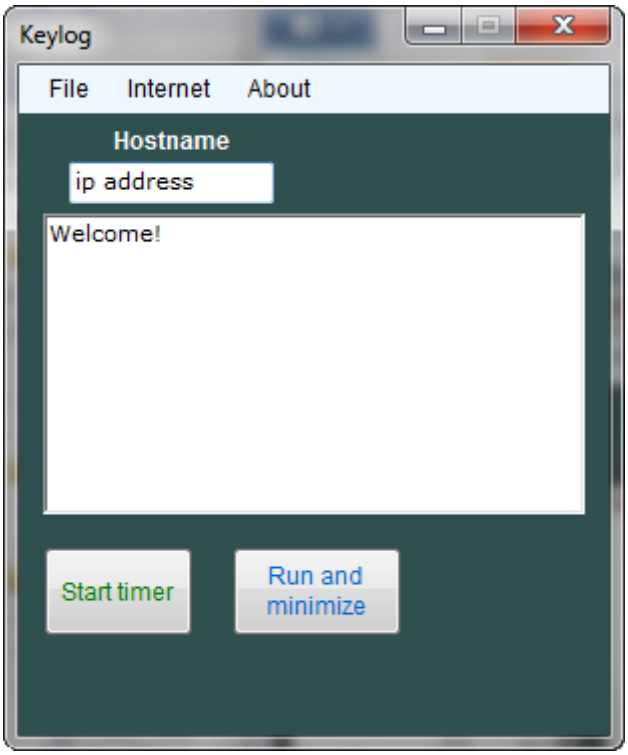

Рис. 8: Интерфейс программы Keylog после запуска

7.3. Замечания по программе

После работы с программой и её разблокирования, файл log.txt будет сохранён и отправлен только в том случае, если таймер был запущен до нажатия кнопки Run. Запуск таймера для отправки файла на удалённый сервер можно также запустить на этом этапе, однако следует учесть, что после запуска таймера все данные записанные в текстовое поле будут удалены и появится сообщение Timer was started.

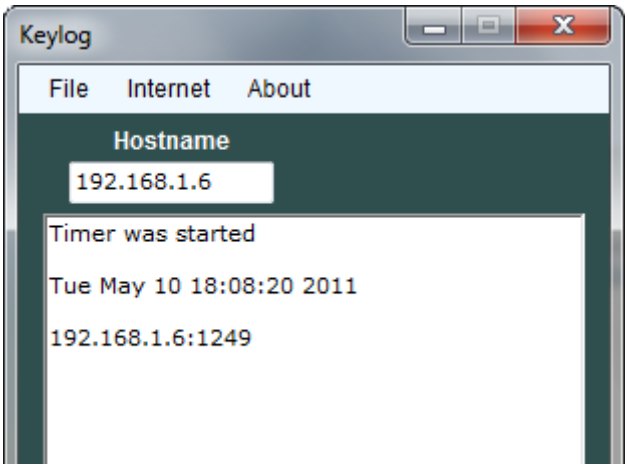

Рис. 9: Информация об отправляемом файле

По истечении таймера, будет показана системная информация об отправляемом файле. Туда входит информация о времени отправки файла в форме дня, месяца, числа и времени, года. Также будет показан ip адрес с виртуальным портом UDP, через который был отправлен файл. Всегда отправляется только текстовый файл log.txt, который должен находиться в корневом каталоге программы. Как только будет снова нажата кнопка RUN, программа свернётся и действие таймера прекратится, а все записанные символы в текстовом поле удалятся. Для прекращения работы таймера необходимо нажать на кнопку “Stop Timer“ во вкладке Internet.

На стороне сервера, файл будет перезаписываться каждый новый раз, после приёма файла log.txt.

7.4. Возможные проблемы

Не поддерживается ввод заглавных букв при включённом CAPS LOCK. Список всех поддерживаемых знаков показан в табл. 3.

Табл. 3: Список поддерживаемых символов

!	4	G	T	g	t
@	5	H	U	h	u
#	6	I	V	i	v
$	7	J	W	j	w
%	8	K	X	k	x
^	9	L	Y	l	y
&	0	M	Z	m	z
*	A	N	a	n	space
.	B	O	b	o	BACKSPACE
,	C	P	c	p	
1	D	Q	d	q	
2	E	R	e	r	
3	F	S	f	s	

7.5. Дальнейшее развитие программы Keylog

Несмотря на то, что программа демонстрирует основные возможности клавиатурного шпиона, этих функций может быть недостаточно. В первую очередь необходимо бы было позаботиться о поддержке многопоточности программы. Это бы позволило выполнение нескольких задач за одно время.

Также бы была возможна замена использующегося протокола иным, более надёжным и более открытым, т.е. который бы не требовал специального сервера для приёма данных, а посылал бы прямо на электронную почту.

Дополнение сервисной информации при перехвате нажатых клавиш в программе только бы упростило работу пользователя анализе текстовых данных.

Расширение до поддержки мультиязычности программы было бы одним из нужных средств дальнейшего развития Keylog.

8. Заключение

Написанная в рамках бакалаврской работы программа Keylog, является всего лишь моделью с целью показать одну из возможных реализаций программы шпиона и продемонстрировать его основные свойства. С помощью такой функции программы как отправка файла, был оценён с точки зрения защиты пользовательский компьютер при работе с сетевыми данными. Протокол, исполняющий данную функцию, полностью подходит к поставленной задаче. Тем самым хотелось бы обратить внимание на проблему шпионских программ и им подобных, а также простоту, с которой они могут быть созданы. Программа Keylog является простой в использовании и демонстрирует основные возможности клавиатурного шпиона.

Данная модель только предполагает демонстрацию основных возможностей клавиатурного шпиона, поэтому, использование данной программы в рамках безопасности не является самоцелью. К тому же, необходимо бы было дополнительно позаботиться о безопасности операционной среды, где используется программа. Однако, как показал результат работы с ОС Windows безопасность которой, отвечает высокому уровню, программа Keylog не может навредить пользователю без его ведома. Так же, результатом этой работы являлось демонстрирование возможностей управлением сетевым трафиком, в частности, стандартных средств Windows, среди которых Firewall, который способен надёжно защитить сетевой несанкционированный входящий и исходящий трафик. В любом случае единственным возможным и удобным способом использования остаётся демонстрация принципа работы клавиатурного шпиона как такового.

Программа обладает также рядом недостатков, которые возникают от однопоточности (линейности) написанной программы, однако наглядно демонстрирует все основные качества клавиатурного шпиона.

Для более удобного анализирования полученных результатов работы программы Keylog, необходимо использовать дополнительные средства программирования, так как после длительной работы с программой достаточно тяжело найти промежуточные данные ввода. Также, более совершенный продукт должен включать в себя мультипоточность, для параллельного действия нескольких операций одновременно.

Развитие же средств защиты позволяет с уверенностью сказать, уже сейчас существуют программы, способные определить кейлоггер любого типа. Однако ни одна, даже самая современная защита этого гарантировать не может. Поэтому необходимо отметить, что наилучшей гарантией от подобного рода программ является, несомненно, только человеческая внимательность.

Используемые приложения

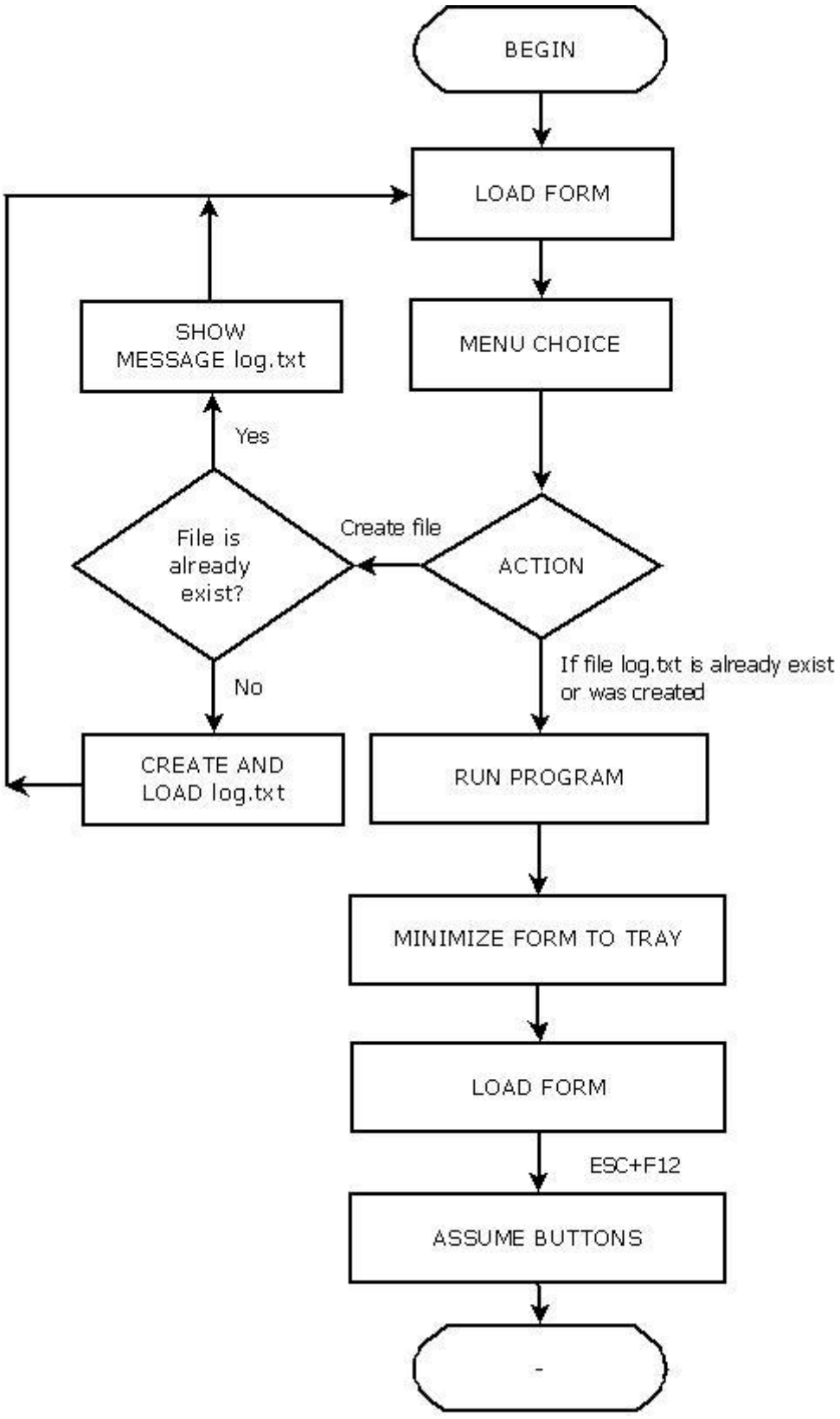

Рис. 10: Блок-схема клавиатурного шпиона

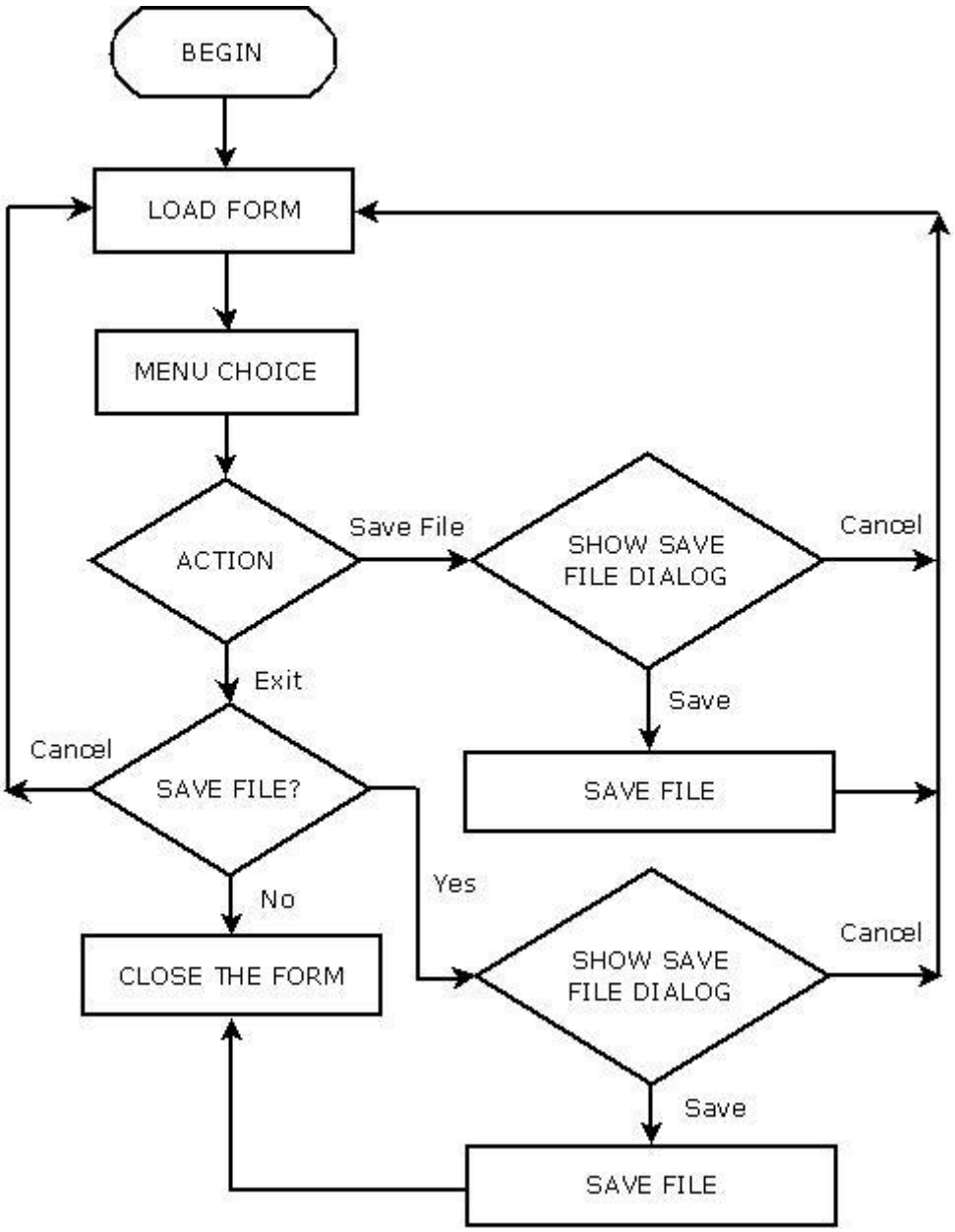

Рис. 11: Блок-схема реализации сохранения файла

Список использующихся сокращений

API	Application Program Interface
ACK	Acknowledge
FTP	File Transfer Protocol
FTPS	File Transfer Protocol + SSL
HTTP	HyperText Transfer Protocol
INF	Setup Information File
INI	Initialization File
RIT	Raw Input Thread
RRQ	Read Request Packet
SHIQ	System Hardware Input Queue
SSH	Secure Shell
SSL	Secure Socket Layer
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
TFTP	Trivial File Transfer Protocol
UDP	User Datagram Protocol
VIQ	Virtualized Input Queue
WRQ	Write Request Packet

Список используемой литературы и цитат

[1] ЗАЙЦЕВ, О.В. Клавиатурные шпионы. *Информационная безопасность* [online]. 2009, 1, [cit.2010-11-30]. Доступно с WWW: <http://www.zoleg.com/secur/articles/keylogger.php>.

[2] РИХТЕР, Джеффри. *Windows для профессионалов*. Санкт-Петербург : Русская Редакция, 2001. 752 s. ISBN 5-272-00384-5.

[3] *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 6.3.2008, last modified on 11.12.2010 [cit. 2010-12-12]. Доступно с WWW: <http://ru.wikipedia.org/wiki/клавиатурный_шпион>.

[4] *Cyberguru.ru* [online]. 2009 [cit. 2010-12-12]. Методы защиты от кейлоггеров. Dostupné z WWW: <http://www.cyberguru.ru/operating-systems/windows-security/keyloggers.html>.

[5] ЗАЙЦЕВ, О.В. *Z-oleg.com* [online]. 2009 [cit. 2010-12-12]. Современные программные и аппаратные клавиатурные шпионы. Доступно с WWW: <http://www.z-oleg.com/secur/articles/keylogger2.php>.

[6] *Driverfiles.ru* [online]. 2010 [cit. 2010-12-12]. Драйверы для Windows XP. Доступно с WWW: <http://www.driverfiles.ru/article/drivers/76.html>.

[7] КАЕВ, А. *Firststeps.ru* [online]. 2009 [cit. 2010-12-12]. INF файлы Windows. Доступно с WWW: <http://www.firststeps.ru/mfc/msdn/r.php?27>.

[8] *Cyberguru.ru* [online]. 2009 [cit. 2010-12-12]. Функция GetAsyncKeyState. Доступно с WWW: <http://www.cyberguru.ru/programming/win32/win32-keyboard-functions-page5.html>.

[9] ТИХОМИРОВ, В.А. *Rsdn.ru* [online]. RSDN Magazine #1. 11.11.2002 [cit. 2010-12-12]. Перехват API-функций в Windows NT/2000/XP. Доступно с WWW: <http://www.rsdn.ru/article/baseserv/IntercetionAPI.xml#E5>.

[10] *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 18.7.2006, last modified on 19.8.2010 [cit. 2010-12-17]. Доступно с WWW: <http://ru.wikipedia.org/wiki/Проактивная_защита>.

[11] *Msdn.microsoft.com* [online]. 12.1.2010 [cit. 2010-12-17]. Keyboard Input. Доступно с WWW: <http://msdn.microsoft.com/enus/library/ms645530(v=VS.85).aspx>.

[12] ЗОЛОТОВ, С. *Devoid.com.ua* [online]. Протоколы Internet. 1998 [cit. 2011-03-24]. Портал программистов. Dostupné z WWW: <http://devoid.com.ua/c-builder/cppbuilder-network-programming/podrobnoe-opisanie-protocola-tcp.html>.

[13] TFTP. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 31.1.2006, last modified on 22.1.2011 [cit. 2011-03-26]. Доступно с WWW: <http://ru.wikipedia.org/wiki/TFTP>.

[14] Udp. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 2.8.2004, last modified on 24.11.2010 [cit. 2011-05-09]. Доступно с WWW: <http://ru.wikipedia.org/wiki/Udp>.

[15] Windows Defender. In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 16.2.2006, last modified on 18.1.2011 [cit. 2011-05-10]. Доступно с WWW: <http://ru.wikipedia.org/wiki/Windows_Defender>.

[16] *Tools.ietf.org* [online]. 1992 [cit. 2011-05-25]. RFC1350. Доступно с WWW: <http://tools.ietf.org/html/rfc1350>.

[17] VIRIUS, Miroslav. *Jazyky C a C++ : Kompletní kapesní průvodce*. Praha : GRADA, 2008. 329 s. ISBN 80-247-1494-9.