ОРИГИНАЛЬНАЯ СТАТЬЯ

УДК 004.056.53(045)

Разработка программного комплекса сканирования открытых баз данных для проведения тестирования на проникновение

Алена Алексеевна Жилина, *студентка факультета информационных технологий и анализа больших данных, Финансовый университет, Москва, Россия*
Alena A. Zhilina*, student, Faculty of Information Technologies and Big Data Analysis, Financial University, Moscow, Russia*
alen.zhilina@yandex.ru

АННОТАЦИЯ

В статье рассмотрено решение задачи поиска уязвимостей в исследуемой информационной инфраструктуре с использованием разработанного программного комплекса сканирования заданного перечня IP-адресов в локальных и глобальных вычислительных сетях на предмет открытых портов баз данных (далее — БД). Ключевыми особенностями разработанного программного решения, отличающими его от аналогов, являются скорость и простота в использовании. Программный продукт обеспечивает выполнение следующих функций: сканирование отдельного узла сети, сканирование подсети по адресу сети и маске подсети, сканирование списка узлов сети из текстового файла, запись результатов сканирования в файл, отображение результатов сканирования в консоли. Для удобного сканирования инструмент можно использовать как консольную утилиту или как Telegram-бот.
Ключевые слова: *корпоративная инфраструктура; несанкционированный доступ; безопасность; базы данных; сканирование; порт; тестирование; Telegram; терминал*

Для цитирования: **Жилина А. А. Разработка программного комплекса сканирования открытых баз данных для проведения тестирования на проникновение.** ***Научные записки молодых исследователей.*** **2022;10(3):47–58.**

ORIGINAL PAPER

Software Development for Scanning Open Databases for Penetration Tests

ABSTRACT

The paper provides the solution to the problem of searching for vulnerabilities in the information infrastructure under study using the software package that developed for scanning a certain list of IP-addresses in local

Научный руководитель: ***Коннова И.Г.,*** *ассистент департамента информационной безопасности, Финансовый университет, Москва, Россия / Scientific supervisor:* ***Konnova I.G.,*** *Assistant of Information Security Department, Financial University, Moscow, Russia.*

and global computer networks for open database ports (DB). The key features of the developed software solution, which differ it from analogues, are speed and simplicity of use. The software provides the following functions: scanning a single host, scanning a subnet by network address and subnet mask, scanning a list of hosts from a text file, saving scan results to a file, displaying scan results in the terminal. A scanner tool as both the console utility and the Telegram-bot can be used for convenient scanning.
Keywords: *corporate infrastructure; unauthorized access; security; databases; scanning; port; testing; Telegram; terminal*

For citation: **Zhilina A. A. Software development for scanning open databases for penetration tests. *Nauchnye zapiski molodykh issledovatelei = Scientific notes of young researchers*. 2022;10(3):47–58.**

Введение

По данным Positive Technologies за 2021 г., целевые кибератаки на корпоративную инфраструктуру компании составили почти 40% (*рис. 1*).

Попытки несанкционированного доступа к конфиденциальной информации осуществляются преимущественно в финансовом секторе и нередко заканчиваются утечкой большого массива данных. Одной из основных причин этого является небезопасная конфигурация систем управления базами данных (далее — СУБД), что является следствием невнимательности или низкой квалификации разработчиков, специалистов по защите информации или сетевых инженеров. Между тем, возможность несанкционированного подключения к одной машине может представлять угрозу как для отдельного рабочего места, так и для локальной сети организации, к которой уязвимый сервер предоставляет доступ, вследствие чего возможен подрыв корпоративной информационной безопасности в целом.

В рамках существующих практик по обнаружению и устранению связанных с этим ошибок имеет место поиск на основе открытых источников (OSINT — Open source intelligence). Следующей ступенью является попытка получения доступа к конфиденциальной информации.

Тестирование на проникновение (penetration testing) необходимо для нахождения в инфраструктуре уязвимых мест, которые потенциально могут быть использованы нарушителями, а также для осознания того, насколько эффективны разработанные политики в области информационной безопасности и нуждаются ли они в совершенствовании.

Одним из методов нахождения потенциальных векторов атаки на инфраструктуру является сканирование узлов сети на предмет наличия открытых портов. Наиболее критичным в данном случае является обнаружение БД, утечка информации из которых может стать серьезной проблемой как для государственных организаций, так и для коммерческих предприятий [1].

Существующие инструменты для проведения такого рода сканирования обладают широким функционалом, однако зачастую недостаточный уровень знаний и навыков пользователей затрудняет использование сканеров в мелкомасштабных и повседневных операциях.

Целью данной работы является разработка программного комплекса, производящего сканирование заданного перечня IP-адресов в локальных и глобальных вычислительных сетях на предмет открытых портов БД, его макетирование и реализация. Программный продукт сможет применяться для составления отчетов по открытым портам на узлах сети, что позволяет обнаружить наиболее незащищенные элементы первого рубежа защиты проверяемой системы, а также проверить статус работы сетевых служб.

Для достижения поставленной цели были определены следующие задачи:

1. Построение алгоритма функционирования программного комплекса.
2. Определение способов применения и сценария действий пользователя.
3. Определение и реализация наиболее интуитивно понятного интерфейса для взаимодействия с пользователем.
4. Разработка архитектуры и реализация программного комплекса в различных вариантах использования.
5. Апробация программного комплекса, а именно, тестирование разработанного приложения на предмет возможных ложных срабатываний.

Рис. 1. **Основные последствия целевых кибератак на российские компании в 2021 г., %**

Источник: URL: https://www.tadviser.ru/index.php/Статья: Потери_организаций_от_киберпреступности.

Обзор сферы и существующих программных решений

Сканирование портов (port scanning) — это способ обнаружения уязвимых узлов в сети путем обращения к различным портам сканируемого объекта (подключенного к сети устройства) или к одному и тому же порту на устройствах.

Различают несколько разновидностей сканирования портов:

1. Горизонтальное сканирование, или сканирование сети — сканирование, при котором запросы поступают на один и тот же порт на разных машинах.
2. Вертикальное сканирование — сканирование, при котором запросы поступают на разные порты на одной и той же машине.

В компаниях, где отлажены процессы ИБ, а также в центрах реагирования и мониторинга используется ПО, позволяющее автоматизировать принятие решений по возникающим инцидентам, но даже при этом многим до сих пор приходится писать скрипты для автоматизации рутинных операций. Процесс проверки портов на локальном рабочем компьютере специалиста по безопасности, на рабочих местах сотрудников, серверах внутренней сети организации, маршрутизаторах должно быть возможным запускать как с определенной периодичностью, так и при необходимости в любой момент времени. Кроме того, это может понадобиться при прохождении компанией внутреннего или внешнего аудита.

Для сканирования портов существует ряд программных продуктов, отличающихся областью применения и способами использования.

Одним из наиболее популярных примеров является Nmap[1]. Nmap — это многофункциональный сканер портов, популярный в сообществе по IT-безопасности. Приложение написано и поддерживается Федором (Fyodor). Nmap — очень гибкий и качественный инструмент, часто используемый для тестирования на проникновение. Он выполняет такие функции, как обнаружение хостов, обнаружение служб и их версий, определение операционной системы (далее — ОС) сканируемого объекта, трассировка сети.

Другим примером является Port Checker[2], бесплатный онлайн- инструмент для проверки наиболее популярных портов: MS SQL (порт 1433), почтовые услуги POP3 (порт 110), IMAP (порт 143), SMTP (порт 25), веб-сервисы HTML (порт 80).

Инструмент, описанный в данной статье, в сравнении с аналогами обладает такими преимуществами, как низкий порог вхождения (для его использования не требуется длительного изучения документации или знания специфических опций) и возможность интеграции с Telegram-ботом. Кроме того, он ориентирован на более узкую область, а именно работу с БД, что делает его удобным для использования при решении задачи, поставленной в данной статье.

Виды баз данных, их основные отличия и особенности

В БД содержится структурированная информация, которой удобно пользоваться. Для выбора нужного типа БД важно понимать, какие именно дан-

[1] Nmap. URL: https://nmap.org/
[2] Port Checker. URL: https://portscaner.ru/

ные будут там храниться и по какому принципу будет удобнее всего работать с ними [2]. Каждый из видов БД ориентирован на решение каких-либо определенных специфических задач.

Основные типы БД:

1. Реляционные. Наиболее популярные представители: MySQL, Oracle DB, PostgreSQL. Данный тип наиболее распространен, в таких БД информация хранится в виде таблиц. В строках содержится описание каждого отдельного свойства объекта, а столбцы нужны для извлечения конкретных свойств из строки. Таблицы могут быть взаимосвязаны.

2. Резидентные. Представители данного типа: Redis, Tarantool, Apache Ignite. Сведения хранятся в оперативной памяти. Данные обрабатываются достаточно быстро, поэтому резидентные БД популярны там, где необходима наиболее минимальная задержка выдачи результатов. Они поддерживают как быстрое написание, так и быстрое чтение. В основном они работают с записями «ключ-значение», но также могут работать со столбцами.

3. Поисковые. Основной пример поисковых БД — ElasticSearch. Этот тип нужен для получения сведений через фильтрацию. Поиск данных можно осуществлять по любому введенному значению, а также по отдельным словам. Есть возможность пользоваться полнотекстовым поиском. Поисковые базы данных достаточно хорошо масштабируются и удобны для хранения журналов и объемных текстовых значений.

4. Документоориентированные. Представители: CouchDB, Couchbase, MongoDB. Одним из недостатков реляционных БД является то, что для извлечения данных нужно объединять их таблицы, а в документоориентированных базах отлично хранится несвязанная информация в больших объемах. Такие БД поддерживают JSON. Для любого ключа можно создать сложное значение и сразу включить всю структуру данных в одну запись. Выборка по запросу может содержать части множества документов без их предварительной полной загрузки в оперативную память.

Выбор языка и среды разработки. Определение интерфейса взаимодействия с пользователем

Как правило, при выборе языка для написания исходного кода программ, выполняющих стандартные задачи системного или сетевого администрирования, выбор встает между языком командной оболочки Bash и полноценными объектно ориентированными языками программирования, такими как Perl, Ruby, Python и т.д. Несмотря на то что с использованием bash-скрипта можно решить довольно обширный круг задач, а работа с разнообразными входными данными, многомерными массивами, сокетами может быть осуществлена на любом из перечисленных языков, для поставленной задачи был выбран Python, обладающий следующим рядом преимуществ:

1. Удобочитаемость и компактность кода.

Наличие четких синтаксических правил делает само написание кода быстрым и гибким, а скрипт — понятным любому IT-специалисту.

2. Наличие большого количества модулей, подключаемых с помощью оператора import.

Большое количество функций и методов поддерживают основные системные протоколы и форматы и легко используются при написании собственного кода. Возможности Python также позволяют написать и подключить собственный модуль.

3. Кроссплатформенность.

Скрипты Python работают и в среде Windows, и в MacOS, и в UNIX, включая FreeBSD и GNU/Linux, в том числе отечественную сертифицированную операционную систему Astra Linux. Этот язык широко используется и на мобильных платформах, таких как Symbian, Android. Написанный на Python, скрипт с большой долей вероятности будет работать на разных платформах, решая схожие задачи, при условии, что код не будет содержать специфических для конкретной операционной системы функций [3].

В качестве среды разработки была выбрана IDE Visual Studio Code. Данный редактор ускоряет работу разработчика, так как одной из его ключевых особенностей является наличие сниппетов (макет/шаблон исходного кода), автоматизирующих ручной труд. Также он имеет интегрированный терминал и систему контроля версий, гибкую систему форматирования и отладчик, а также делает возможной совместную работу над проектом.

При выборе формата пользовательского интерфейса было определено два наиболее удобных. При постоянном использовании терминала специалисту по информационной безопасности, занимающемуся тестированием на проникновение, предоставляется возможность использования

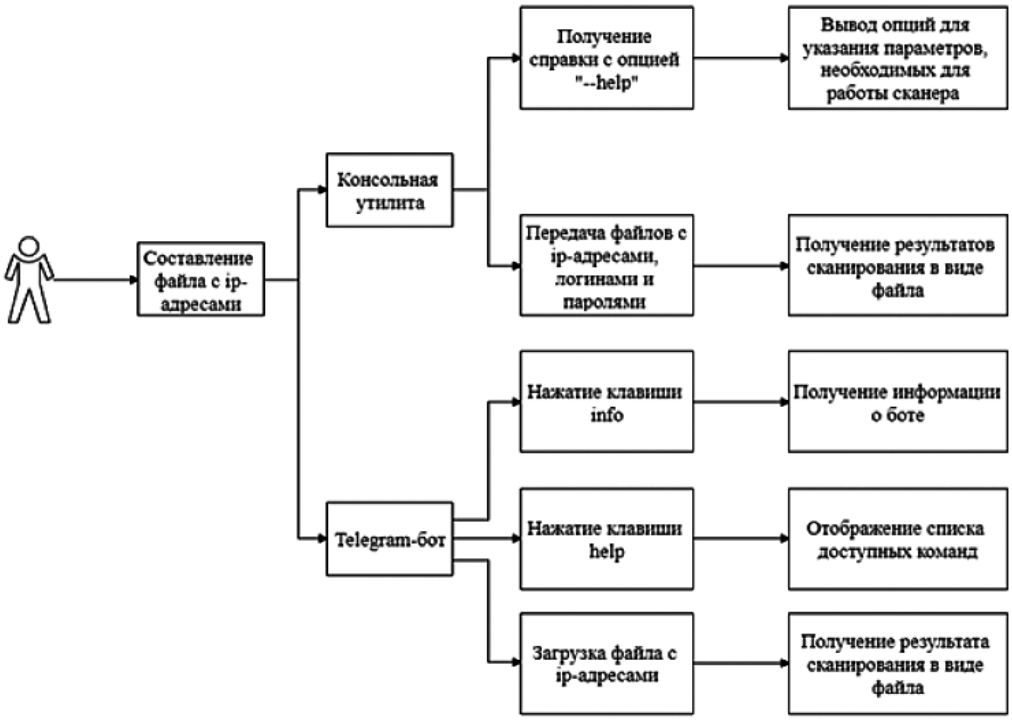

Рис. 2. **Диаграмма вариантов использования**

Источник: составлено автором.

программного комплекса в качестве консольной утилиты. В случае, если при проведении сканирования нет необходимости следить за подробным ходом работы программы и более важным является получение результата с возможностью своевременного реагирования на инциденты, утилита может быть использована как Telegram-бот, предоставляющий подробное описание использования и справку по командам.

В целом, предполагаемые сценарии действий пользователя могут быть представлены на *рис. 2*.

Разработка программного комплекса сканирования открытых баз данных

Разработанный программный комплекс осуществляет сканирование в автоматическом режиме, кроме того, для повышения оперативности работы специалиста по информационной безопасности реализована интеграция с Telegram-ботом.

Основной модуль программного комплекса выполняет следующие задачи:

1. Взаимодействие с пользователем, запрос исходных данных.
2. Перевод диапазонов IP-адресов в формат, удобный для сканирования.
3. Попытка подключения к портам баз данных при нахождении их в перечне IP-адресов.
4. Формирование выходного файла.
5. Предоставление пользователю результатов сканирования.

Так как в файле с адресами должна быть возможность помимо единичных адресов указывать диапазоны адресов в разных форматах (например: 192.168.0.1/24, 192.168.0.1–254, 192.168.0–100.1–254 и т.д.), то в файл ip_all.txt сохраняются автоматически все IP-адреса, полученные из этих диапазонов.

Для хранения IP-адресов, на которых был обнаружен хотя бы один порт, используется файл csv.txt, где сохраняется данная информация в виде

<IP-адрес>:<открытые порты>, например, 104.244.53.18:27017 и 190.202.40.53:5432,27017.

Программная часть состоит из[3]:

- scan_v4.py — функции сканирования;
- hacking.py — функции для выполнения подключения к БД и записи результатов в log-файл;
- bot.py — Telegram-бот.

В сборку также включены:

- GeoIP2-City.mmdb — локальная БД для поиска соответствия региона конкретному IP-адресу;

[3] Злой сканер, как написать инструмент что всегда под рукой. URL: https://codeby.net/threads/zloj-skaner-kak-napisat-instrument-chto-vsegda-pod-rukoj.75878/

- install.sh — файл установки программного комплекса;
- ip.txt — подборка IP-адресов для тестирования;
- logins.txt, top-usernames-shortlist.txt — выборка популярных логинов для тестирования;
- password.txt и xato-net-10-million-passwords-10.txt — выборка популярных паролей для тестирования;
- requirements.txt — файл зависимостей для среды python.

В качестве входной информации выступают IP-адреса, выборка логинов и паролей в виде текстовых файлов, а также файл с настройками setting.conf для конфигурации. В файле настроек имеется возможность задания портов по умолчанию для конкретных БД, задержку по времени между сканированием адресов, название и путь к GeoIP-БД в формате *.mmdb для поиска информации по конкретному IP-адресу и возможность очистки log-файла (log.txt) перед сканированием.

При необходимости можно использовать свой список логинов и паролей. При нахождении портов БД производится попытка подключения к ним по заданным словарям логинов и паролей. При успешном подключении происходит запрос следующего перечня имеющихся БД на сканируемой системе: MongoDB, Elasticsearch, MySQL, PostgreSQL.

Полученная информация записывается в log-файл. Работа программного комплекса выполняется посредством консоли ОС Linux.

Алгоритм функционирования представлен на *рис. 3*.

Пошаговая методика применения программного комплекса представлена в *таблице*.

Запуск и работа с программным продуктом посредством терминала

При использовании в качестве консольной утилиты возможны два способа установки.

1 способ:

```
chmod +x install.sh
bash ./ install.sh
```

2 способ:

```
pip3 install -r requirements.txt
```

Запустить на сканирование можно следующей командой:

```
python3 scan_v4.py -i ip.txt -l logins.txt -p password.txt
```

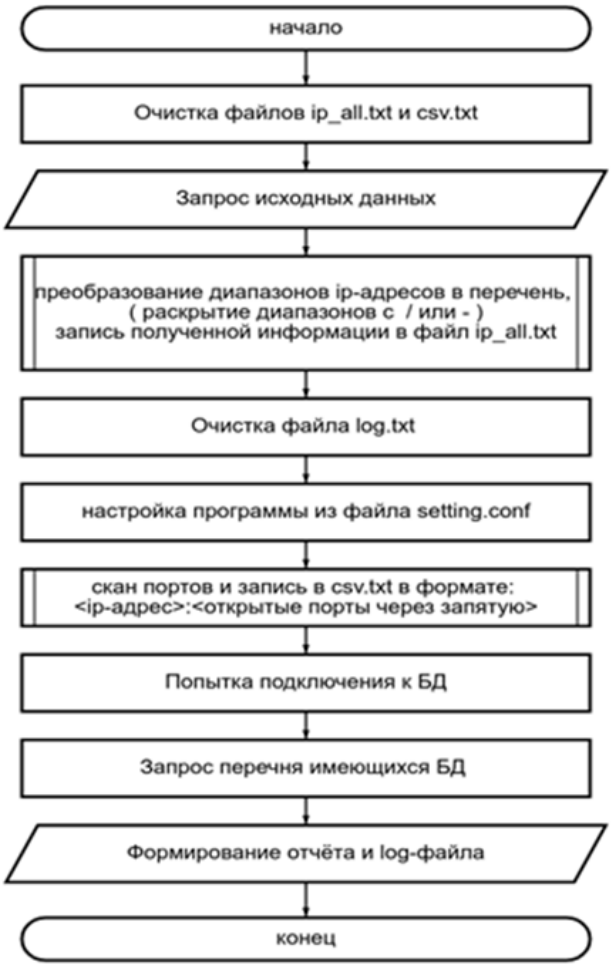

Рис. 3. **Алгоритм функционирования программного комплекса**

Источник: составлено автором.

```
54.38.242.81
122.51.82.64
101.132.147.55
127.0.0.1
85.192.12.42
64.90.48.204
39.101.137.227
```

Рис. 4. **Пример содержания входного файла**

Источник: авторский скриншот.

Во входной файл возможна передача как перечня IP-адресов, так и подсети по адресу сети и маске подсети. Разные IP-адреса и маски подсетей указываются с новой строки (*рис. 4*).

Результат сканирования записывается в выходной log-файл, в котором указывается IP-адрес проверяемого хоста, принадлежность IP-адреса к стране и региону, номера открытых портов,

Таблица

Пошаговая методика применения программного комплекса

Номер шага	Действия
Шаг 1	Начало
Шаг 2	Очистка файлов ip_all.txt и csv.txt
Шаг 3	Открытие файла с IP-адресами, предоставленного пользователем, преобразование диапазонов IP-адресов в перечень, запись полученной информации в файл ip_all.txt
Шаг 4	Очистка файла log.txt (если задано в настройках)
Шаг 5	Сканирование IP-адресов путем установки tcp-соединения с использованием SOCK_STREAM
Шаг 6	Запись в файл csv.txt при нахождении хотя бы одного открытого порта
Шаг 7	Запись в файл log.txt информации о тех адресах, на которых были найдены открытые порты, поиск IP-адреса по БД GeoIP2-City.mmdb для определения страны, к которой он принадлежит
Шаг 8	Поиск IP-адреса по БД GeoIP2-City.mmdb для определения страны, к которой он принадлежит
Шаг 9	Подключение к БД по данным файла csv.txt (IP-адрес и порт)
Шаг 10	Запись результатов в log.txt (тип найденной БД, подошедшая связка логин/пароль)
Шаг 11	Запрос перечня имеющихся БД при успешном подключении
Шаг 12	Вывод информации в виде log-файла
Шаг 13	Конец

Источник: составлено автором.

название типа и таблиц в БД[4]. Пример такого файла приведен на *рис. 5*.

В программе также реализован цветной вывод в консоль во время сканирования и во время попытки подключения к БД.

Процесс выполнения сканирования в терминале представлен на *рис. 6–8*.

Запуск и работа с программным продуктом посредством Telegram-бота

Программный продукт может быть развернут как Telegram-бот и работать постоянно как сервис на выделенном сервере. Предварительно необходимо создать бота в Telegram через @BotFather, выбрать отображаемое имя и активировать его.

```
--------------------
ip: 104.244.53.18
country: United States
READ_ME_TO_RECOVER_YOUR_DATA
admin
config

open_port: 27017 --- MongoDB

--------------------
ip: 101.132.147.55
country: China

open_port: 3306 --- MySQL
DATA_RECOVERY
admin
config

open_port: 27017 --- MongoDB

--------------------
ip: 39.101.137.227
country: Hong Kong
read_me
think_tank

open_port: 9200 --- Elasticsearch
```

Рис. 5. **Выходные данные в виде log-файла**

Источник: составлено автором.

[4] Основы применения Python в администрировании Linux. URL: https://www.ibm.com/developerworks/ru/library/l-python/index.html

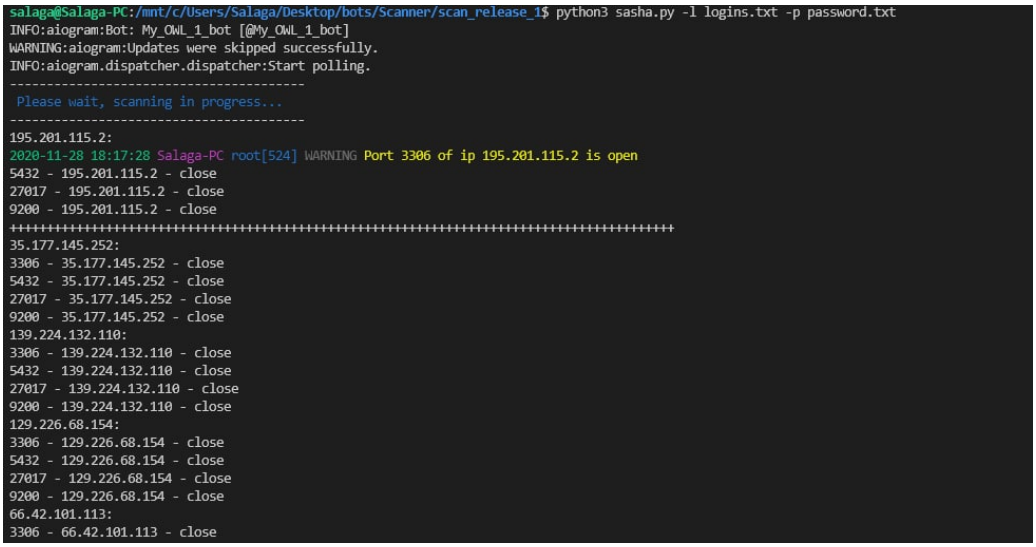

Рис. 6. **Начало сканирования в терминале**

Источник: составлено автором.

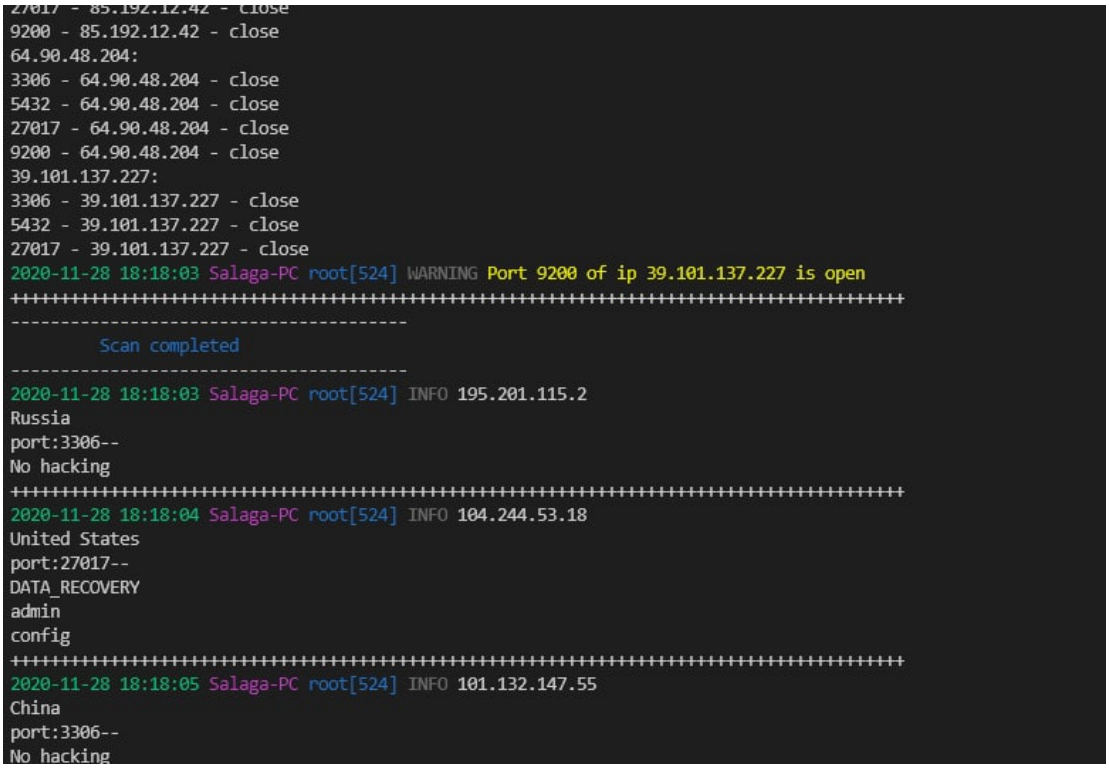

Рис. 7. **Процесс сканирования в терминале**

Источник: составлено автором.

После этого будет выдан токен для подключения сервиса сканирования к созданному боту.

При запуске программы в Telegram-боте появится приветственное окно с описанием системы (*рис. 9*).

При вводе команды /start ботом будут отправлены инструкции по использованию (*рис. 10, 11*).

Работа пользователя с ботом строится на отправке боту текстовых сообщений или файлов с IP-адресами для сканирования. Процесс выполнения работы ботом не виден пользователю, что может быть более удобным в повседневной работе, когда фильтрация лишней информации экономит время сотрудника. В качестве результата пользователь получает файл log.txt, аналогич-

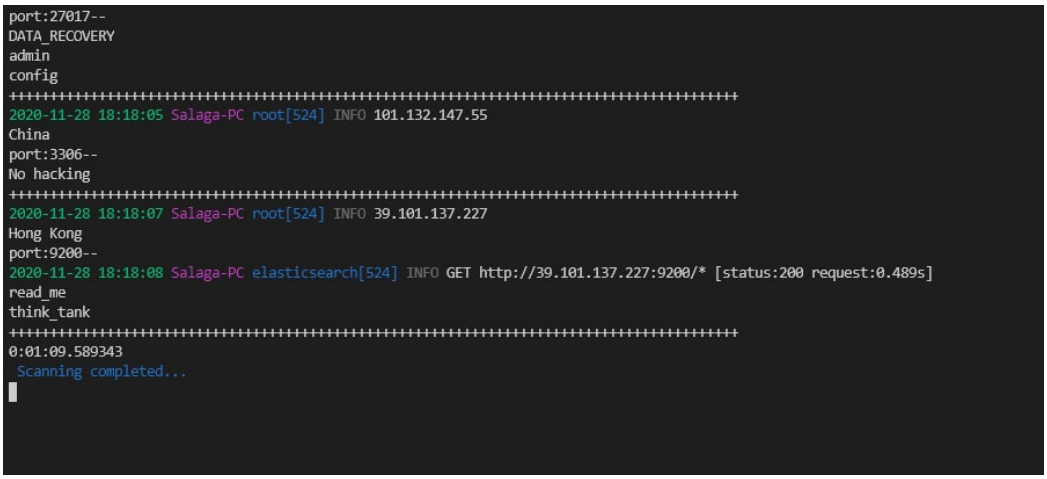

Рис. 8. **Завершение сканирования в терминале**

Источник: составлено автором.

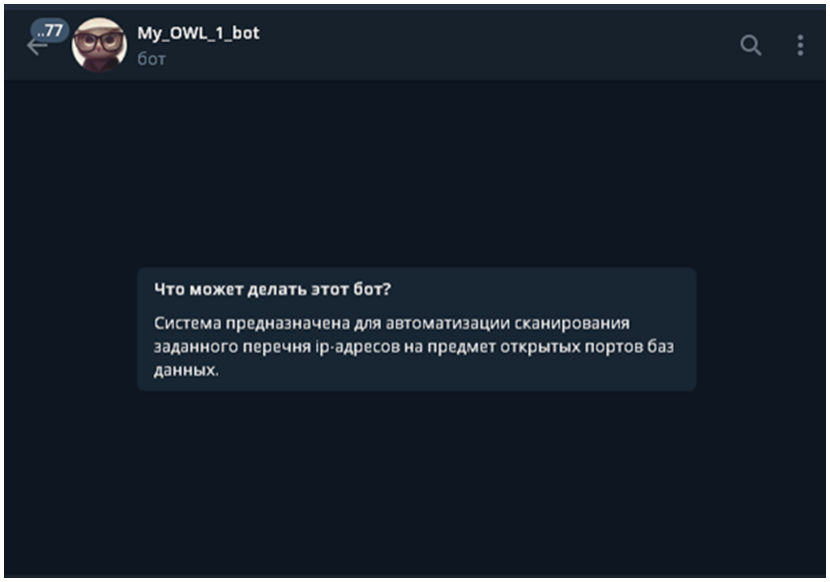

Рис. 9. **Приветственное окно**

Источник: составлено автором.

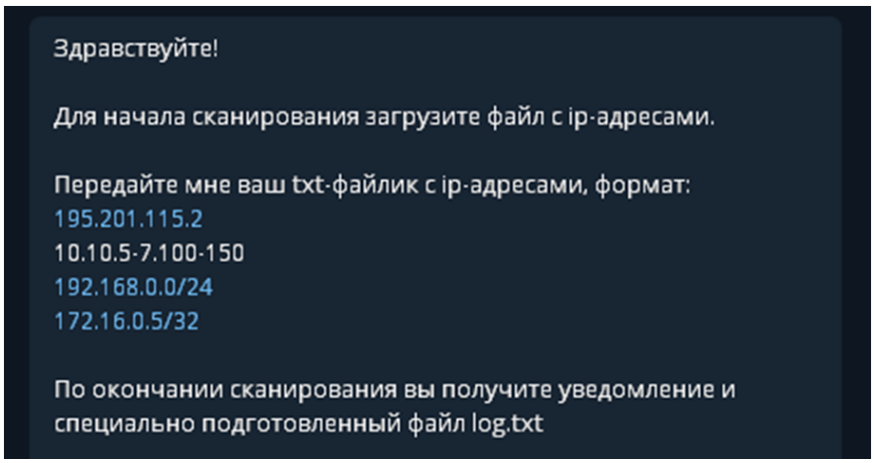

Рис. 10. **Сообщение о начале работы**

Источник: составлено автором.

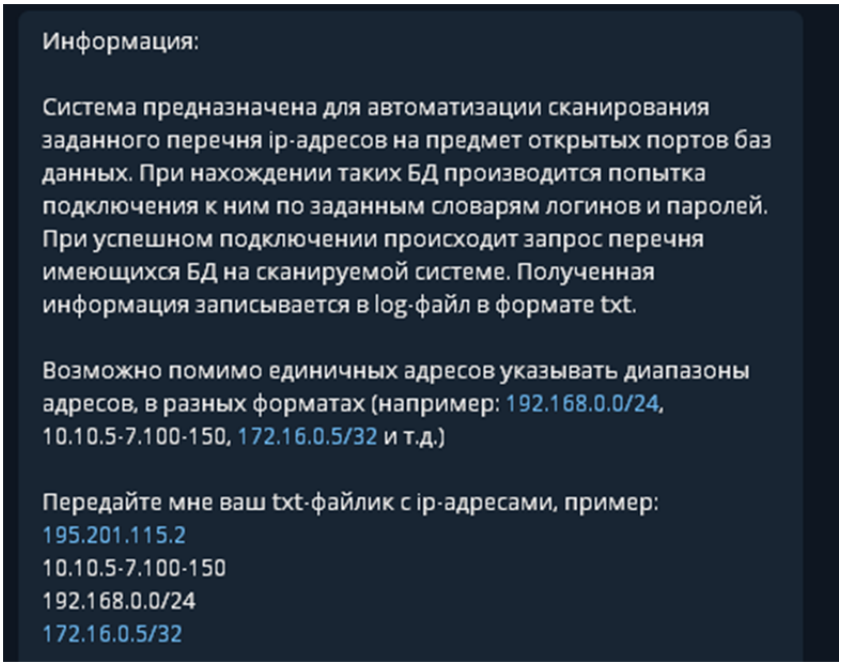

Рис. 11. **Справочная информация**

Источник: составлено автором.

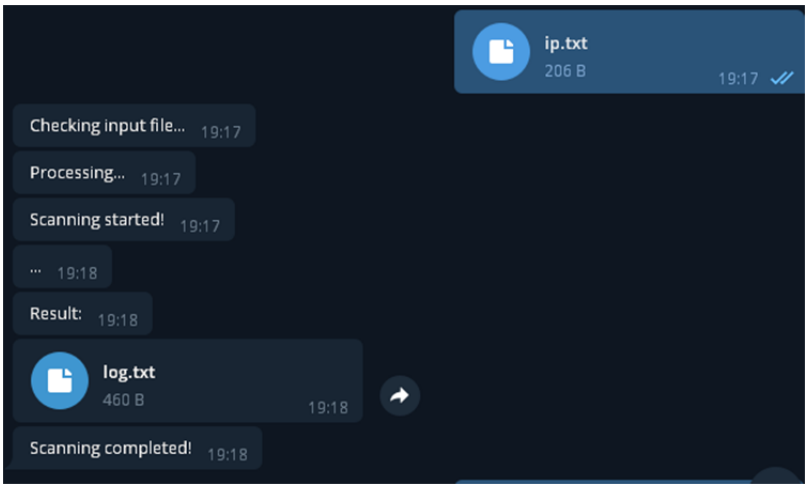

Рис. 12. **Процесс взаимодействия пользователя с Telegram-ботом**

Источник: составлено автором.

ный получаемому при консольном сканировании (*рис. 12*).

Тестирование на проникновение, аудит и способы защиты

Существует ряд уязвимостей в СУБД, которые могут привести не только к реализации несанкционированного доступа к информации, хранящейся в БД, но и к компрометации всей сети в целом.

Основным типом атак, которые наиболее активно осуществляют злоумышленники, является BruteForce — подбор пароля исходя из наиболее часто используемых слов. Перебор производится только по значениям, заданным в конкретном словаре. Часто данной атаке предшествует атака User enumeration — перечисление пользователей. Ее проведение возможно на серверах, которые поддерживают старые механизмы аутентификации (CVE-2012–5615). Результатом сканирования User enumeration является информация о том, какие пользователи существуют в БД, что значительно сокращает пул пользователей для BruteForce`а.

Используя известное имя пользователя (например, root, который присутствует практически всегда) с любым паролем, можно подключиться к базе, повторяя подключение порядка 300 раз.

```
Usage: sudo PK_script.sh [OPTION]...

All options are required.
  -i    name of your network interface on the server
  -P    name of the service you want to hide
  -p    port number corresponding to this service
  -k    three space separated port numbers that are a knock sequence

Example:
  sudo PK_script.sh -i ens32 -P SSH -p 22 -k 1100 2200 3300
```

Рис. 13. **Справка по работе со скриптом для настройки port-knocking**

Источник: составлено автором.

После этого можно получить все пароли пользователей, перебрать их и подключаться уже с легитимным паролем.

Это только один из сценариев проведения атаки на БД. Разработанный программный комплекс в данном случае может служить инструментом предотвращения подобных атак. Ряд стандартов, на соответствие требованиям которых компания может проходить проверку (например, PCI DSS), регламентируют использование нестандартных портов для популярных служб. Сканер портов может быть использован для нахождения СУБД, расположенных на портах по умолчанию, что будет сигнализировать о необходимости принятия мер по защите такой службы.

Альтернативным методом защиты портов является фильтрация техникой port-knocking. [4] Ее суть состоит в определении ключевой последовательности портов (от 3 и более), которая будет известна только работникам организации, внутренним сервисам, скриптам, которым необходимо предоставить доступ к БД. При подключении пользователю необходимо «простучать» заданную последовательность портов с помощью утилиты knock или telnet. После чего демон службы knockd предоставляет доступ на определенный период для ip-адреса, с которого был произведен «стук». Блокировка и открытие портов осуществляется с помощью правил iptables. В репозитории автора статьи на github можно найти скрипты для автоматической настройки port-knocking на серверах с ОС CentOS и Ubuntu версий 16, 18, 19, 20[5].

Для использования скрипта необходимо поместить файл.sh в удобную пользователю директорию на сервере, а далее установить права на выполнение файла (на примере Ubuntu 20):

chmod +x ./PK_script_Ubuntu19–20.sh

Первоначально скрипт можно запустить с опцией -h для ознакомления со справкой по необходимым параметрам (*рис. 13*).

Соответственно, для настройки pork-knocking скрипту необходимо указать название интерфейса, порт на котором будет спрятан, наименование сервиса, номер порта и последовательность портов для «стука».

После запуска в соответствии с заданными параметрами будет:

1. Установлена служба knockd и iptables-persistent (если не установлена):

apt-get install -y knockd iptables-persistent

2. Сохранен бэкап конфигурационного файла knockd:

mv /etc/knockd.conf /etc/knockd.conf.bak

3. В новый конфигурационный файл записываются параметры и правила iptables со значениями, заданными пользователем:

echo -e “[options]\n\tUseSyslog\n\tInterface = ${iface}\n[${protocol}]\n\tsequence = ${port1},${port2},${port3}\n\tseq_timeout = 15\n\ttcpflags = syn\n\tstart_command = /sbin/iptables -I INPUT -s%IP% -p tcp —dport ${port} -j ACCEPT\n\tcmd_timeout = 1800\n\tstop_command = /sbin/iptables -D INPUT -s%IP% -p tcp —dport ${port} -j ACCEPT” > /etc/knockd.conf

Кроме того, есть возможность самостоятельно установить время, в течение которого должна быть введена последовательность «стука» (параметр start_command в секундах), и время,

[5] Репозиторий со скриптами для port-knocking. URL: https://github.com/ellyzing/useful_scripts/tree/main/port-knocking

после которого произойдет автозакрытие интерфейса для входящих подключений (параметр stop_command в секундах).

1В файле /etc/default/knockd параметр START_KNOCKD устанавливается в значение 1:

sed -i «s/START_KNOCKD=0/START_KNOCKD=1/g» «/etc/default/knockd»

4. Выбранный порт на указанном интерфейсе закрывается и правила сохраняются:

iptables -A INPUT -m state —state ESTABLISHED, RELATED -j ACCEPT

iptables -A INPUT -p tcp —destination-port «$port» -i «$iface» -j DROP

iptables-save > /etc/iptables/rules.v4

5. Служба knockd добавляется в автозагрузку и запускается:

systemctl enable knockd.service

systemctl start knockd

Теперь на клиентской стороне перед подключением по ssh необходимо выполнить:

knock <ip or domain name of your server> 1100

knock <ip or domain name of your server> 2200

knock <ip or domain name of your server> 3300

ssh <username>@ <ip or domain name of your server>

Для приведенной выше конфигурации последовательности команд необходимо выполнять в течение 15 секунд. Автозакрытие произойдет через 30 минут.

Для закрытия ssh:

knock <ip or domain name of your server> 3300

knock <ip or domain name of your server> 2200

knock <ip or domain name of your server> 1100

Выводы

Программный комплекс обеспечивает выполнение функций тестирования отдельного узла сети, подсети по адресу сети и маске подсети, сканирования списка узлов сети из текстового файла, записи результатов сканирования в файл, отображения результатов сканирования в консоли, подключения к различным базам данных с последующим выводом доступной информации.

Преимуществом предлагаемого средства является возможность оперативного информирования заинтересованных специалистов о результатах тестирования посредством Telegram-сообщений. Проведенная апробация показала возможность его применения для оперативной проверки средств вычислительной техники на предмет соответствия требованиям (политикам) информационной безопасности.

Разработанное программное средство является перспективным, доверенным программным комплексом с возможностью функционирования в двух удобных для специалиста по безопасности режимах.

Список источников

1. Marsh N. Nmap 6 Cookbook: The Fat Free Guide to Network Security Scanning 6th Edition. 198 p. ISBN 13:978–1507781388.
2. Полищук Ю.В. Базы данных и их безопасность. Учебное пособие. Студентам ССУЗов. М.: Инфра-М; 2021. 210 с. ISBN 978–5–16–016151–8.
3. Мартелли А., Рейвенскрофт А. Python. Полное описание языка. М.: Вильямс; 2020. 896 с. ISBN 978–5–6040723–8–7.
4. Жилина А.А. Практика защиты ssh-авторизации. Обнаружение и предотвращение атак на ssh. Информационная безопасность в современном мире, 2020:61–78. ISBN 978–5–406–08816–6.

References

1. Marsh N. Nmap 6 Cookbook: The fat free guide to network security scanning 6th ed. 198 p. ISBN 13:978–1507781388.
2. Polishchuk Yu.V. Databases and their security. Tutorial. College students. Moscow: Infra-M; 2021. 210 p. ISBN 978–5–16–016151–8. (In Russ.).
3. Martelli A., Ravenscroft A. Python. Full language description. Moscow: Williams; 2020. 896 p. ISBN 978–5–6040723–8–7. (In Russ.).
4. Zhilina A.A. The practice of protecting SSH authorization. Detection and prevention of attacks on SSH. Information security in the modern world, 2020:61–78. ISBN 978–5–406–08816–6. (In Russ.).