РАЗРАБОТКА СЦЕНАРИЕВ БЕЗОПАСНОСТИ ДЛЯ СОЗДАНИЯ УЯЗВИМЫХ ВИРТУАЛЬНЫХ МАШИН И ИЗУЧЕНИЯ МЕТОДОВ ТЕСТИРОВАНИЯ НА ПРОНИКНОВЕНИЯ

С. И. Штеренберг*, А. И. Москальчук, А. В. Красов

Санкт-Петербургский государственный университет телекоммуникаций
им. проф. М. А. Бонч-Бруевича,
Санкт-Петербург, 193232, Российская Федерация

*Адрес для переписки: stas.shterenberg.89@mail.ru

Аннотация—В статье демонстрируется концепция, при которой построение лаборатории для тестирования на проникновение осуществляется при помощи специальной программы. Программа представляет собой набор скриптов, которые конфигурируют систему в соответствие с заданным пользователем сценарием. Благодаря элементам рандомизации сценариев данное решение позволяет развернуть сразу несколько учебных задач для группы студентов, используя только один образ виртуальной машины. Основная идея заключается в том, что настройка и создание уязвимой цели происходит непосредственно перед выполнением самой учебной задачи. Т. е. изначально виртуальная машина является базовым образом Ubuntu Linux, не обладающим каким-либо набором уязвимостей. Главной особенностью предлагаемого решения является то, что содержание скриптов описывает не один вариант конфигурации системы, а сразу несколько, образуя сценарии с элементами рандомизации. Иначе говоря, имея базовый образ Ubuntu Linux и набор вышеупомянутых скриптов, можно создать различные друг от друга задачи для десятка студентов.

Ключевые слова—информационная безопасность, тестирование на проникновение, учебная лаборатория, виртуализация, CTF.

Информация о статье
УДК 004.56
Язык статьи – русский.
Поступила в редакцию 02.03.2021, принята к печати 24.03.2021.

Ссылка для цитирования: Штеренберг С. И., Москальчук А. И., Красов А. В. Разработка сценариев безопасности для создания уязвимых виртуальных машин и изучения методов тестирования на проникновения // Информационные технологии и телекоммуникации. 2021. Том 9. № 1. С. 47–58. DOI 10.31854/2307-1303-2021-9-1-47-58.

DEVELOP SECURITY SCRIPTS TO CREATE VULNERABLE VIRTUAL MACHINES AND LEARN PENETRATION TESTING TECHNIQUES

S. Shterenberg*, A. Moskalchuk, A. Krasov

The Bonch-Bruevich Saint-Petersburg State University of Telecommunications,
St. Petersburg, 193232, Russian Federation

*Corresponding author: stas.shterenberg.89@mail.ru

Abstract—The article demonstrates the concept of building a laboratory for penetration testing using a special program. The program is a set of scripts that configure the system in accordance with a user-defined script. Thanks to the elements of script randomization, this solution allows you to deploy several educational tasks at once to a group of students using only one virtual machine image. The basic idea is that the setup and creation of a vulnerable target occurs just before the execution of the learning task itself. Those, the virtual machine is initially a basic Ubuntu Linux image that does not have any set of vulnerabilities. The main feature of the proposed solution is that the content of the scripts describes not one variant of the system configuration, but several at once, forming scripts with elements of randomization. In other words, having a basic Ubuntu Linux image and a set of the scripts, you can create different tasks for a dozen students.

Keywords—information security, penetration testing, training laboratory, virtualization, CTF.

Article info
Article in Russian.
Received 02.03.2021, accepted 24.03.2021.

For citation: Shterenberg S., Moskalchuk A., Krasov A.: Develop security scripts to create vulnerable virtual machines and learn penetration testing techniques // Telecom IT. 2021. Vol. 9. Iss. 1. pp. 47–58 (in Russian). DOI 10.31854/2307-1303-2021-9-1-47-58.

Введение

С каждым годом востребованность в проведении тестирования на проникновение заметно возрастает. Данная необходимость обусловлена тем, что современные компании сильно заинтересованы в поддерживании достойного уровня информационной безопасности (далее ИБ). Утечка критичных данных может за собой нести как существенные финансовые, так и репутационные риски. Однако, для построения наилучшей защиты, необходимо оценить её первоначальный уровень и отталкиваться от полученных результатов и возможных рисков. Одной из лучших общепринятых практик является как раз тестирование на проникновение, которое представляет собой симуляцию кибератаки на информационную инфраструктуру компании. Тестирование на проникновение может предоставить реалистичную оценку текущего уровня ИБ и продемонстрировать, какие потенциальные риски несет проникновение злоумышленника в информационную инфраструктуру [1, 2, 3]. В большинстве случаев тестирование на проникновение может являться как хорошим началом для построения безопасной инфраструктуры, так и для поддерживания уровня ИБ в дальнейшем.

Данная проблема натолкнула на создание такого решения, благодаря которому студент или преподаватель может использовать один образ ВМ для развертывания сразу нескольких учебных задач, различных друг от друга. Это может позволить избежать процесса поиска и установки новых целей в имеющуюся учебную лабораторию и существенно сократить процент несамостоятельного выполнения работ. Важно отметить, что предлагаемое решение нацелено в первую очередь на студентов университетов, которые только начинают знакомство с методами тестирования на проникновение.

Описание предлагаемого решения

Основная идея заключается в том, что настройка и создание уязвимой цели происходит непосредственно перед выполнением самой учебной задачи. Т. е. изначально ВМ является базовым образом Ubuntu Linux, не обладающим каким-либо набором уязвимостей. Конфигурирование ВМ и создание учебной задачи осуществляется при помощи набора скриптов, которые заранее загружаются из внешнего источника [4]. Главной особенностью предлагаемого решения является то, что содержание скриптов описывает не один вариант конфигурации системы, а сразу несколько, образуя сценарии с элементами рандомизации. Иначе говоря, имея базовый образ Ubuntu Linux и набор вышеупомянутых скриптов, можно создать различные друг от друга задачи для десятка студентов.

На рис. 1 (ниже) можно увидеть высокоуровневую модель, которая иллюстрирует пример разворачивания учебной лаборатории. Как видно из рис. 1, от пользователя требуется минимальное количество возможных действий от момента установки до перехода к практике [5]. На первом этапе необходимо иметь лишь средство виртуализации и образ ВМ Ubuntu Linux. Дальнейшим шагом является загрузка конфигурационных скриптов на целевой сервер. После этого можно инициализировать основной скрипт конфигурации и приступить к выполнению учебных задач.

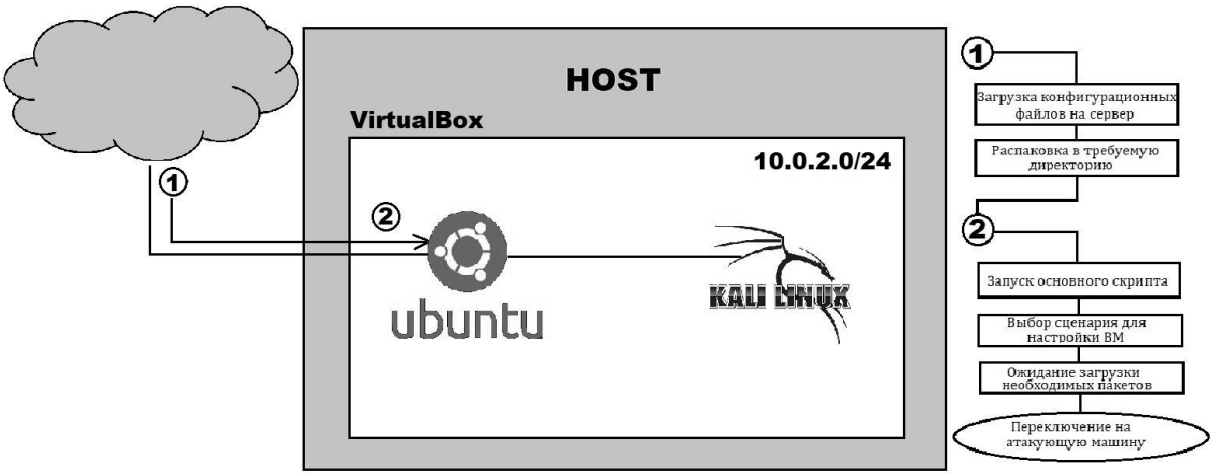

Рис. 1. Пример разворачивания учебной лаборатории

Скрипты написаны на языке программирования bash и вместе образуют своего рода цельную программу, отвечающую за настройку виртуальной машины. Взаимодействие пользователя осуществляется только с одним скриптом – *masterscr.sh*, основными функциями которого являются следующие операции:

- имитация пользовательского интерфейса;
- выбор необходимого сценария для конфигурации системы;
- отправка команд нижестоящим скриптам.

После запуска основного скрипта конфигурации – *masterscr.sh,* пользователю будет предложено выбрать один из возможных сценариев для дальнейшей настройки виртуального сервера. На данный момент, в процессе разработки находятся 3 возможных сценария:

1) Поиск флага;
2) Повышение привилегий;
3) Эксплуатация уязвимостей CVE.

После выбора нужного сценария инициализируется запуск нижестоящих скриптов, которые в свою очередь полностью конфигурируют систему. Весь процесс настройки происходит без участия студента, а весь попутный текстовый вывод фиксируется в отдельный журнал [6]. Однако, в случае возникновения ошибки программа закончит свою работу и сообщит пользователю, из-за чего возникла та или иная ошибка.

Большинство нижестоящих скриптов связаны между собой, благодаря чему формируются отдельные конфигурационные сценарии. Содержаение скриптов и время настройки может различаться в зависимости выбора конкретного сценария, но в общем виде они могут иметь следующие содержание:

- установка и настройка сервисов;
- добавление пользователей и назначение привелегий;
- открытие портов;
- настройка правил межсетвого экранирования;
- генерация флагов;
- изменение конфигурационных файлов.

На рис. 2 продемонстрирован процесс инциализации основного скрипта и последующая настройка виртуального сервера[1]. Как видно, пользователем был выбран сценарий под номером один – поиск флага. После его запуска, программа сообщает, что необходимо время на настройку сервера. После завершения настройки можно увидеть сообщение, что все процессы были завершены успешно.

```
root@student_server:~# ./masterscr.sh


Please select a script for system configuration:
   1) Capture the Flag script
   2) Escalation of privileges script
   3) CVE exploitation script
   0) Exit


Write number here:
1
OK! Please wait...


All right. Congratulations !
```

Рис. 2. Инициализация программы и выбор сценария для конфигурации системы

Демонстрация

После того как сервер был настроен, можно начинать целевую эксплуатацию. В данной части статьи будет продемонстрирован процесс выполнения задачи со стороны студента, его возможные действия и используемые инструменты. Предполагается, что студент предварительно не знает об особенностях конфигурации целевой системы, т. е. выполняет тестирование «черного ящика». В конкретном случае «атакующей» ВМ будет выступать ОС Kali Linux.

Ранее был выбран сценарий поиска флага, процесс инициализации которого был продемонстрирован на рис. 2. Соответственно, основная цель, которая стоит перед студентом, найти способ получить сгенерированный флаг.

Приступая к выполнению задачи, первое, что необходимо сделать – это раскрыть живые хосты в сегменте сети и найти IP-адрес целевого сервера. Для этого можно использовать утилиту Nmap, а именно её опцию *-sP* (пинг-разведка Nmap). Данный параметр обрабатывает каждый IP-адрес в заданном диапазоне сети. В случае если узел включен и настроен для ответа на запросы *ping*, он выдаст ICMP ответ. Из рис. 3 видно, что в ходе сканирования было обнаружено два живых хоста [7]. Также известно, что на IP-адресе 10.0.2.14 находится ВМ Kali Linux. Соответственно, IP-адрес целевого сервера – 10.0.2.15.

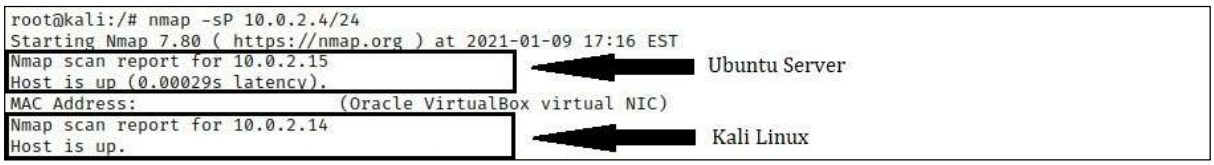

Рис. 3. Результаты сканирования Nmap с использованием опции ping-sweep

[1] PCI DSS. PCI Security Standards Council [Электронный ресурс]. URL: http://www.pcisecuritystandards.org

После определения IP-адреса «цели» следующим шагом может быть сканирование TCP портов. Данный шаг может предоставить данные об открытых, закрытых, фильтрованных и нефильтрованных сетевых портах, а также основную информацию о работающих службах и установленных версиях. Перечисленная информация может стать ценной для получения удаленного доступа на сервер и обнаружения флага. Исходя из этого, было выполнено сканирование всех портов TCP при помощи команды *nmap –Pn –sV 10.0.2.15 –p 1-65535* [8]. Опция *–Pn* обрабатывает все активные хосты в заданном диапазоне, а *–sV* исследует открытые порты для определения информации о сервисах и их версиях. На данное сканирование необходимо время, однако оно может предоставить ценную информацию для понимания атакуемой системы. На рис. 4 продемонстрированы результаты, полученные в ходе сканирования Nmap. На отображенных данных видно, что на целевом хосте открыты порты 80 и 2222. Также, благодаря расширенному сканированию Nmap стало известно, что на данных портах работают HTTP и SSH.

Кроме этого, на рис. 4 можно также увидеть результаты сканирования TCP портов перед запуском конфигурационного скрипта (рис. 3). Как видно из результатов, ни один из 65353 портов не был обнаружен в ходе сканирования, так как система имела полностью базовую конфигурацию.

Имея информацию об IP-адресе целевой системы и базовые сведения о работающих сервисах, студент может приступать к дальнейшему исследованию системы. Конечно, можно сразу приступить к попытке взлома SSH методом перебора. Однако, имя пользователя и пароль на данный момент остаются неизвестными, из-за чего подбор по имеющимся словарям может отнять большое количество времени и не дать ощутимых результатов. Исходя из этого, студент может попытаться найти какие-либо «подсказки», которые помогут получить удаленный доступ к системе и вополнить поставленную задачу.

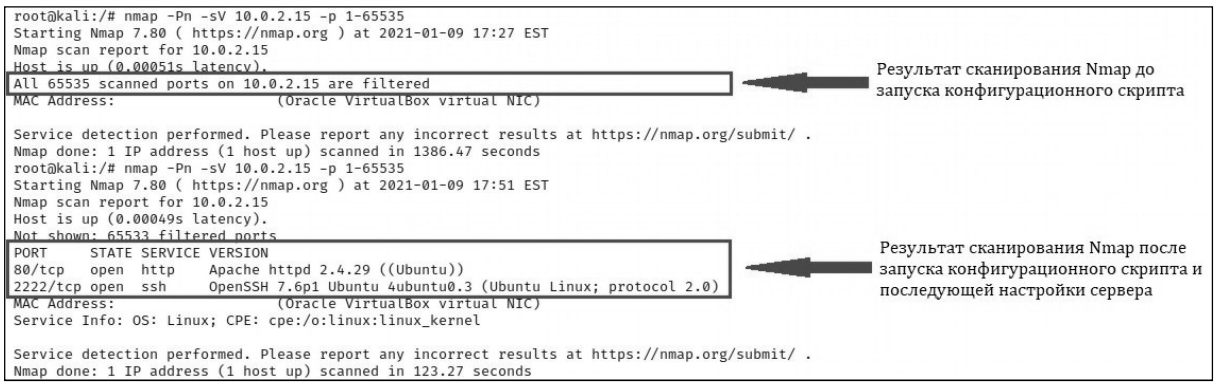

Рис. 4. Результаты сканирования TCP портов до и после запуска конфигурационного скрипта

Из результатов сканирования Nmap было известно, что на целевом хосте работает веб-сервер Apache. Из этого следует, что после перехода по IP-адресу в браузере в качестве ответа можно получить веб-страницу с атакуемого сервера. Как видно из рис. 5, после перехода по IP-адресу можно увидеть форму авторизации. Однако, после открытия кода страницы стало видно, что в нем

присутсвует также несколько закомментированных хэш-значений MD5, которые не были отображены во время загрузки страницы.

Рис. 5. Веб-страница полученная от целевого сервера

MD5 – это алгоритм, позволяющий получить своего рода «отпечаток» исходной строки. Он работает таким образом, чтобы из получившегося отпечатка нельзя было восстановить исходное сообщение [9]. По большому счету расшифровать MD5 невозможно, однако существуют такие сервисы, посредством которых можно найти заранее посчитанные значения. После использования специального сервиса были получены соответствующие значения обнаруженным ранее хэшам:

95bce48e24d8a8f8cb4cb729ee5a73ae – www.password.com
84cc0e5c525dc728e1769ad6663341c8 – www.test.com
23eb143ed83f3c2c4a98e84ad196f07c – www.qwerty.com
7c1767b30512b6003fd3c2e618a86522 – www.example.com

Исходя из того, что целевой веб-сервер работает локально, в файл */etc/hosts* на ВМ Kali Linux были добавлены обнаруженные доменные имена. После перехода по одному из адресов (www.example.com) были отображены следующие аутентификационные данные:

Login: lxdadmin
Password: 8888*******

На данном этапе стал известен логин пользователя, учетная запись которого, вероятно, присутсвует на целевом веб-сервере. Хотя пароль был получен в усеченном виде, данной информации уже достаточно, чтобы студент мог реализовать атаку на SSH методом перебора. В системе Kali Linux предустановлено несколько специализированных инструментов, которые позволяют автоматизировать данный процесс. Одним из наиболее известных

и функциональных инструментов перебора паролей является Hydra. Синтаксис управления Hydra достаточно прост, однако необходимо иметь файл со словарем паролей, которые будут использоваться во время перебора. Конечно, можно использовать известные, уже готовые словари, содержащие наиболее распространенные и уязвимые пароли, но также можно создать и собственный. Хорошим решением в данном случае будет являться утилита Crunch. Crunch позволяет создавать словари паролей со всевозможными комбинациями и перестановками в соответствии с заданными критериями.

Исходя из полученных данных, был создан словарь паролей, длина которых равняется одиннадцати символам. Кроме этого, на этапе сбора информации стало ясно, что первые цифры пароля – 8888. Поэтому, чтобы сократить итоговый объем, была использована опция *–t*, которая позволяет задать требуемый паттерн для всех паролей. На рис. 6 можно увидеть команду, которая использовалась для создания необходимого словаря [10].

Зная логин пользователя, IP-адрес целевого сервера, нужный порт и имея словарь паролей, можно приступать к взлому SSH методом перебора. В конкретном случае использовалась команда *hydra –V –f –t 4 –l –P /home/kali/Documents/wordlist.txt ssh://10.0.2.15:2222/*. Благодаря опции *–V* происходит текстовый вывод на экран всех перебираемых паролей. Опция *–f* завершает действие программы, если целевой пароль был найден, а опции *–l* и *–P* указывают программе, что происходит перебор паролей из заданного файла. На рис. 6 можно увидеть процесс атаки на SSH и успешное завершение программы, когда целевой пароль был найден. На основе полученных данных был выполнен успешный вход на сервер от пользователя *lxdadmin.*

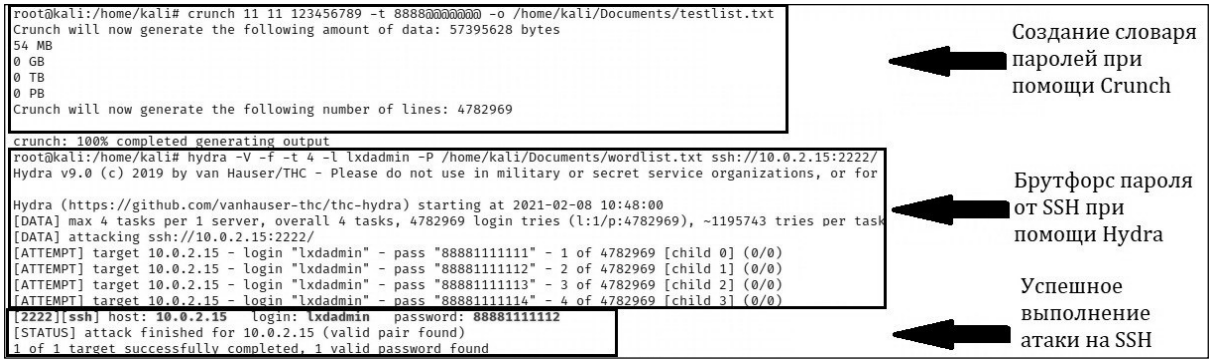

Рис. 6. Создание словаря паролей и атака на SHH методом перебора

Первое, что может выполнить студент, после удаленного входа в систему – это исследовать уровень доступа пользователя и имеющиеся файлы в домашней директории. Из рис. 7 можно увидеть, что вход был осуществлен в директорию */home/lxdadmin*, в которой находится файл *key.txt*, который, очевидно, является целевым флагом. Однако права на чтение и редактирование данного файла есть только у пользователя *root*. Исходя из этого, появляется необходимость повысить уровень доступа и найти способ раскрыть содержание файла *key.txt*. *Lxdadmin* является локальным пользователем без административных привилегий, однако, как видно из рисунка ниже, он также является частью группы *lxd* [11].

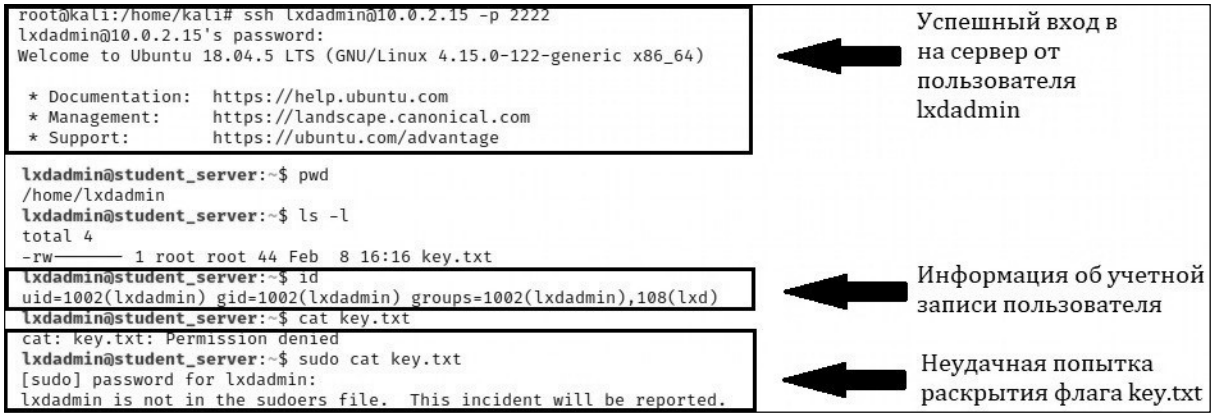

Рис. 7. Успешный вход на сервер от пользователя lxdadmin

LXD – это системный менеджер контейнеров. Он предлагает пользовательский интерфейс, похожий на виртуальные машины, но использующий вместо этого контейнеры Linux. Существует уязвимость, посредством которой член локальной группы *lxd* может мгновенно повысить привилегии до уровня *root* в операционной системе хоста. Единственное необходимое требование для реализации данной уязвимости – это доступ к учетной записи пользователя, которая является членом группы *lxd*. Локальный пользователь из данной группы может монтировать корневой каталог хоста в контейнер, что позволит ему читать файлы, которые ранее были доступны только для привилегированных пользователей. Для эксплуатации данной уязвимости и последующего повышения привилегий необходимо выполнить следующие действия:

1) Загрузить образ Alpine Linux на ВМ Kali Linux;
2) Выполнить сценарий *«build -alpine»*, который собирает последний образ Alpine Linux в виде сжатого tar-файла;
3) Перенести tar-файл с Alpine Linux на целевой сервер;
4) Импортировать изображение в *lxd* на целевом сервере;
5) Инициализировать изображение внутри нового контейнера;
6) Смонтировать каталог root внутрь контейнера.

После загрузки и сборки образа Alpine Linux на ВМ Kali Linux, был запущен простой HTTP сервер, при помощи которого необходимый tar-файл был доставлен на целевой хост в директорию */tmp*. Для создания контейнера и монтирования в него корневого каталога были выполнены следующие действия:

```
# импорт изображения
  lxc image import ./alpine-v3.13-x86_64-20210207_1141.tar.gz --alias myimage
  # создание контейнера с доступом к жесткому диску хоста
  lxc init myimage container1 -c security.privileged=true
  # монтирование /root в контейнер
  lxc config device add container1 mydevice disk source=/ path=/mnt/root recursive=true
  # взаимодействие с контейнером
  lxc start container1
  lxc exec container1 /bin/sh
```

Листинг 1. Монтирование корневого каталога в контейнер

После получения доступа к контейнеру, был выполнен переход в директорию */mnt/root*, где находились все файлы целевого сервера. Как видно из рис. 8, вход был осуществлен от пользователя *root*, у которого есть доступ на чтение и запись файла *key.txt*. Таким образом, была выполнена эскалация привилегий путем эксплуатации известной уязвимости в *LXD*. Это позволило получить сгенерированный флаг и выполнить учебную задачу.

Рис. 8. Эскалация привелегий и получение флага key.txt

Заключение

В заключение данной статьи важно отметить, что демонстрационная часть затронула только один вариант конфигурации системы, которая может быть получена после запуска программы. Как видно из полученных результатов, в ходе выполнения лишь одной учебной задачи студент должен был применить знания в области Linux систем, поиска и эксплуатации уязвимостей, а также применения специализированных инструментов тестирования на проникновение.

На данный момент, в процессе активной разработки находятся сценарии по следующим направлениям: поиск флага, повышение привилегий, эксплуатация уязвимостей CVE. Помимо разработки новых сценариев также принимаются во внимание способы оптимизации процесса установки и доставки конченому пользователю. Наиболее приоритетными направлениями являются вариативность сценариев и отказоустойчивость во время конфигурации целевой системы. Разработка данного проекта осуществляется в рамках программы TEMPUS (*Trans-European Mobility Programme for University Studies*).

Литература

1. Андрианов В. И., Юркин Д. В., Стасюк В. В. Разработка пентест лаборатории // Наукоемкие технологии в космических исследованиях Земли. 2019. Т. 11. № 4. С. 56–64. DOI 10.24411/2409-5419-2018-10279.

2. Красов А. В., Штеренберг С. И., Москальчук А. И. Методология создания виртуальной лаборатории для тестирования безопасности распределенных информационных систем // Вестник Брянского государственного технического университета. 2020. Т. № 3 (88). С. 38–46. DOI 10.30987/1999-8775-2020-3-38-46.

3. Шива П., Замм А., Буду Д., Йохансен Д., Аллен Л., Хериянто Т., Али Ш. Kali Linux. Тестирование на проникновение и безопасность. СПб.: Питер, 2020. 448 с. ISBN 978-5-4461-1252-4.

4. Штеренберг С. И., Штеренберг И. Г. Вероятностные методы построения элементов самообучения адаптивных информационных систем // Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. 2016. № 1. С. 53–56.

5. Сахаров Д. В., Левин М. В., Фостач Е. С., Виткова Л. А. Исследование механизмов обеспечения защищенного доступа к данным, размещенным в облачной инфраструктуре // Наукоемкие технологии в космических исследованиях Земли. 2017. Т. 9. № 2. С. 40–46.

6. Катасонов А. И., Штеренберг С. И., Цветков А. Ю. Оценка стойкости механизма, реализующего мандатную сущностно-ролевую модель разграничения прав доступа в операционных системах семейства gnu linux // Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. 2020. № 2. С. 50–56 DOI 10.46418/2079-8199_2020_2_8

7. Wilhelm Th. Professional penetration testing : creating and learning in a hacking lab / by edition Matthew Neely. 2nd ed. Waltham, MA : Syngress, 2013. ISBN 9780124046184.

8. Херцог Р., О'Горман Д., Ахарони М. Kali Linux от разработчиков. СПб.: Изд-во Питер, 2019. 320 с. ISBN 978-5-4461-0548-9.

9. Виткова Л. А., Ахрамеева К. А., Грузинский Б. А. Использование геометрических хеш-функций в информационной безопасности // Известия высших учебных заведений. Технология легкой промышленности. 2017. Т. 37. № 3. С. 5–9.

10. Милосердов А. В. Тестирование на проникновение с помощью Kali Linux 2.0 / под ред. Д. А. Гриднева. WebWare.biz, 2015. 348 с.

11. Цветков А. Ю. Исследование существующих механизмов защиты операционных систем семейства LINUX // Актуальные проблемы инфотелекоммуникаций в науке и образовании (АПИНО 2018) : VII Международная научно-техническая и научно-методическая конференция. Сб. науч. ст. в 4-х т. Санкт-Петербург, 28 февраля – 01 марта 2018 года / Под ред. С. В. Бачевского. СПб.: Изд-во СПбГУТ, 2018. Т. 1. С. 657–662. ISBN 978-5-89160-167-3.

References

1. Andrianov V. I., Yurkin D. V., Stasyuk V. V. Development pentest laboratories // H&ES RESEARCH. 2019. Vol. 11. Iss. 4. pp. 56–64 (in Russian). DOI 10.24411/2409-5419-2018-10279.

2. Krasov A., Shterenberg S., Moskal'chuk A. Virtual laboratory creation for distributed information system safety testing // Bulletin of Bryansk State Technical University. 2020. Vol. Iss. 3 (88). pp. 38–46 (in Russian). DOI 10.30987/1999-8775-2020-3-38-46.

3. Shiva V. N. P., Samm A., Boodoo D., Johansen G., Allen L., Heriyanto T., Sh. Ali Kali Linux 2018: Assuring Security by Penetration Testing. Birmingham. Packt Publishing 2018. First published in the English language under the title «Kali Linux 2018. ISBN 978-1789341768 (in English).

4. Shterenberg S. I., Shterenberg I. G. Probabilistic methods for constructing elements of self-adaptive information systems // Vestnik of St. Petersburg State University of Technology and Design. Series 1. Natural and technical science. 2016. Iss. 1. pp. 53–56 (in Russian).

5. Sakharov D. V., Levin M. V., Fostach E. S., Vitkova L. A. Research of mechanisms of the protected access problem to cloud data storage // H&ES RESEARCH. 2017. Vol. 9. Iss. 2. pp. 40–46 (in Russian).

6. Katasonov A. I., Shterenberg S. I., Tsvetkov A. Yu. Estimation of the resistance of a mechanism implementing a mandatory essential-role model of division of access rights in operating systems of the GNU LINUX family // Vestnik of St. Petersburg State University of Technology and Design. Series 1. Natural and technical science. 2020. Iss. 2. pp. 50–56 (in Russian). DOI 10.46418/2079-8199_2020_2_8

7. Wilhelm Th. Professional penetration testing : creating and learning in a hacking lab / by edition Matthew Neely. 2nd ed. Waltham, MA : Syngress, 2013. ISBN 9780124046184.

8. Hertzog R., O'Gorman J., Aharoni M. Kali Linux Revealed. Cornelius, NC: Offensive Security. 2017. 346 p. (in English). ISBN 978-0997615609.

9. Vitkova L. A., Akhrameeva K. S., Gruzinskiy B. A. Geometric hashing usage in information security // The News of higher educational institutions. Technology of Light Industry. 2017. Vol. 37. Iss. 3. pp. 5–9 (in Russian).

10. Miloserdov A. V. Testirovanie na proniknovenie s pomoshch'yu Kali Linux 2.0 / pod red. D. A. Gridneva. WebWare.biz, 2015. 348 s. (in Russian).

11. Tcvetkov A. Research of Existing Mechanisms of Protection of Linux Operation Systems // Aktualnye problemy infotelekommunikacij v nauke i obrazovanii (APINO 2018) : VII Mezhdunarodnaya

nauchno-tekhnicheskaya i nauchno-metodicheskaya konferenciya. Sb. nauch. st. v 4-h t. Sankt-Peterburg, 28 fevralya – 01 marta 2018 goda / Pod red. S. V. Bachevskogo. SPb.: Izd-vo SPbGUT, 2018. T. 1. pp. 657–662 (in Russian). ISBN 978-5-89160-167-3.

Штеренберг Станислав Игоревич
кандидат технических наук, доцент, доцент кафедры Санкт-Петербургского государственного университета телекоммуникаций им. проф. М. А. Бонч-Бруевича, stas.shterenberg.89@mail.ru

Москальчук Андрей Игоревич
студент Санкт-Петербургского государственного университета телекоммуникаций им. проф. М. А. Бонч-Бруевича, andreymoskalchyk0812@gmail.com

Красов Андрей Владимирович
кандидат технических наук, доцент, заведующий кафедрой Санкт-Петербургского государственного университета телекоммуникаций им. проф. М. А. Бонч-Бруевича, krasov@inbox.ru

Shterenberg Stanislav I.
candidate of engineering sciences, docent, associate professor, The Bonch-Bruevich Saint-Petersburg State University of Telecommunications, stas.shterenberg.89@mail.ru

Moskalchuk Andrey I.
student, The Bonch-Bruevich Saint-Petersburg State University of Telecommunications, andreymoskalchyk0812@gmail.com

Krasov Andrey V.
candidate of engineering sciences, docent, head of department, The Bonch-Bruevich Saint-Petersburg State University of Telecommunications, krasov@inbox.ru