УДК 004.056.5 Вестник УрФО № 1(51) / 2024, с. 60–65

Кузьмина У.В., Михайлова О.Е., Афанасьев Ю.П. DOI: 10.14529/secur240108

РАЗРАБОТКА МОДУЛЯ КИБЕРПОЛИГОНА ДЛЯ ЗАЩИТЫ ВЕБ-СЕРВИСОВ ОТ АТАК НА ОСНОВЕ НЕЙРОСЕТИ

В данной статье разрабатывается модуль киберполигона для защиты веб-сервисов от атак на основе нейросетей. Этот модуль анализирует входящий HTTP-трафик для выявления пакетов, содержащих аномалии и попытки атак, далее генерирует предупреждение, которое заносится в логи и отправляется администратору сети. Также был разработан модуль для построения виртуальных сетевых лабораторий, что позволит исследователям и инженерам создавать, тестировать и анализировать как новые, так и старые методы защиты в контролируемых условиях.

Ключевые слова: *информационная безопасность, киберполигон, нейросети, веб-сервисы, фаервол.*

Kuzmina U.V., Mikhaylova O.E., Afanasyev Yu.P.

DEVELOPMENT OF A CYBERPOLYGON MODULE TO PROTECT WEB SERVICES FROM ATTACKS BASED ON A NEURAL NETWORK

In this article, a cyberpolygon module is being developed to protect web services from attacks based on neural networks. This module analyzes incoming HTTP traffic to identify packets containing anomalies and attack attempts, then generates a warning that is logged and sent to the network administrator. A module has also been developed for building virtual network laboratories, which will allow researchers and engineers to create, test and analyze both new and old protection methods in controlled conditions.

Keywords: *information security, cyberpolygon, neural networks, web services, firewall.*

Введение

В настоящее время широко применяют киберполигоны для моделирования сетевой инфраструктуры предприятия и ее тестирования на проникновения. Также киберполигоны могут использоваться для обучения правильного построения сетей, методологий атак и тактик защиты.

Задачей модуля киберполигона является полноценная имитация сети и составляющих ее устройств. В возможности полигона входят создание виртуальных машин на основе имеющихся шаблонов, создание новых шаблонов, построение связей между машинами и сохранение построенной сети с возможностью быстрой загрузки [1].

На текущий момент существуют и эксплуатируются полигоны Минкомсвязи России, Ernst & Young, Cyberbit Range, Cyber Range Hub, Cybersecurity Nexus (CSX). К основным минусам можно отнести необходимость в поддержании работы большого кол-ва реального оборудования, ограниченность сценариев атаки и защиты, сложность в масштабировании и реализации новой инфраструктуры и сценариев.

Популярность веб-приложений постоянно растет, поскольку с их помощью предприятия и корпорации размещают множество своих сервисов. Однако, организации попрежнему сталкиваются с уязвимостями в веб-приложениях. Для защиты веб-приложений используются различные инструменты, например WAF (Web Application Firewall), а также методологии по написанию безопасного кода и обработки входных данных.

Российский рынок Web Application Firewall, как и многие другие сферы, находится в периоде турбулентности. Изменения, вызванные уходом зарубежных поставщиков, ещё не закончились: отечественные вендоры активно осваивают освободившиеся пространства и развивают функциональные возможности своих систем, чтобы удовлетворить ожидания новых заказчиков. При этом значительная часть потенциальных клиентов вообще не использует WAF, предпочитая, очевидно, универсальные решения, такие как NGFW. Мысль о необходимости специализированной защиты веб-трафика зачастую приходит к таким компаниям слишком поздно — уже после взлома.

Однако не все атаки и уязвимости возможно исключить благодаря этим методам и инструментам. Зачастую требуется анализ специалистом информационной безопасности. Однако, обрабатывать весь объем входящих данных является затруднительной задачей.

Задачей модуля для защиты является защита веб-сервисов от типовых атак на веб-приложения и предупреждение администратора веб-сервиса. Модуль должен перехватывать HTTP-пакеты и анализировать с помощью нейросети на наличие вредоносных нагрузок и нетипичное содержание и структуру HTTP-пакетов. При обнаружении таких нагрузок модуль генерирует предупреждение, которое заносится в логи и отправляется администратору сети [2].

Перед началом разработки программной части модуля для построения виртуальных сетевых лабораторий были выдвинуты следующие требования к функционалу:

1. Возможность быстро развернуть приложение с модулем.

2. Поддержка создания пользовательских виртуальных машин в среде построения лаборатории.

3. Создание виртуальных сетевых интерфейсов у машин.

4. Создание изолированных виртуальных сетей для объединения виртуальных интерфейсов машин.

5. Удаленное подключение к виртуальным машинам.

6. Настройка виртуальных машин (ограничение используемой оперативной памяти и процессорного времени).

7. Сохранение конфигураций лабораторий, виртуальных сетей и машин.

8. Возможность создавать лаборатории с предустановленной конфигурацией сети и машин.

9. Наличие графического интерфейса для управления лабораторией.

Для реализации модуля построения виртуальных сетевых лабораторий с учетом описанных требований использовались следующие технологии и подходы:

• Контейнеризация - для изоляции процессов и ресурсов виртуальных машин друг от друга и хост-системы, а также для обеспечения быстрого развёртывания [3].

• Виртуализация устройств – для работы виртуальных машины выбрана система QEMU/KVM. Эта система позволяет эмулировать работу большинства устройств и операционных систем. Также некоторые устройства и операционные системы могут обрабатываться QEMU/KVM напрямую средствами рабочего контейнера, что увеличивает производительность [4].

• Виртуальные сетевые адаптеры – для создания виртуальных сетей и соединения виртуальных машин между собой и с внешними сетями. Использовались встроенные воз-

можности QEMU/KVM в связке с возможностями приложения контейнеризации [5].

• Языки программирования – для написания кода модуля использовался Python, так как позволяет работать на операционных системах Windows и Linux.

• Для хранения параметров и конфигураций лабораторий, виртуальных машин и сетей использовались xml-схемы.

• API – для управления виртуальными сетевыми лабораториями было разработано API на Python, с помощью которого можно автоматически создавать, настраивать и удалять виртуальные машины и сети.

Для достижения поставленных целей было разработано приложение для моделирования конфигурации сети. В приложении доступны следующие функции:

1. Создание и удаление виртуальных машин

2. Соединение виртуальных машин виртуальными сетевыми мостами

3. Удаленное подключение к виртуальным машинам

Приложение имеет графический интерфейс для создания, перемещения машин по экрану, созданию и удалению связей между машинами, а также текущий статус машины и связей. Данный интерфейс представлен на рис. 1.

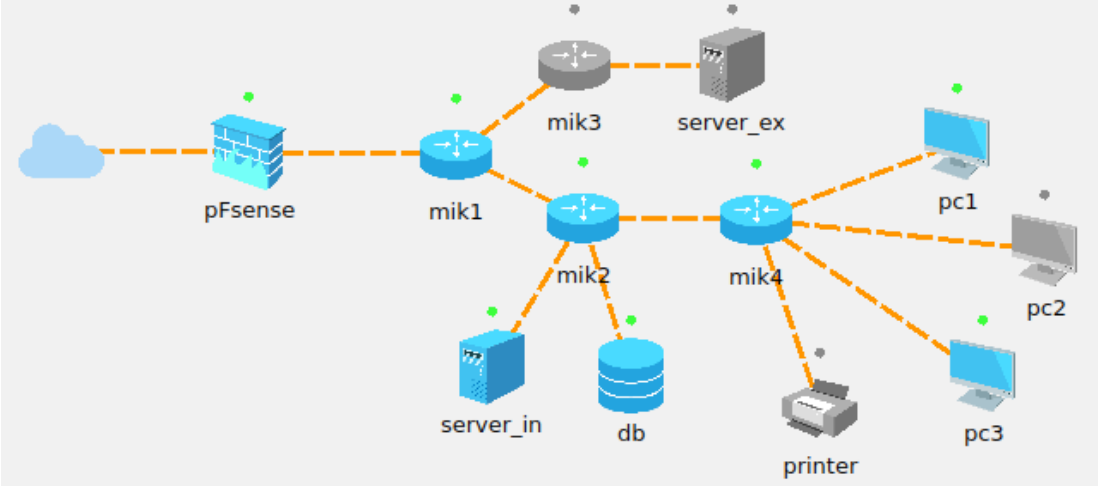

Рис. 1. Графический интерфейс приложения

Разрабатываемый модуль для защиты веб-сервисов разрабатывается на основе нейросети. Это поможет обнаруживать не только аномальные и вредоносные пакеты на основе существующих атак, но также и новые типы атак, еще не опубликованные исследователями и компаниями [6].

Приложение включает в себя механизм перехвата пакетов для анализа, модуль обученной нейросети для анализа входящего трафика, а также механизм логирования аномальных/вредоносных пакетов.

Для решения задачи анализа HTTP-трафика на предмет аномалий и атак можно использовать различные типы нейронных сетей, в зависимости от конкретной задачи и доступных данных.

Один из наиболее распространенных типов нейронных сетей для обнаружения аномалий в данных – это **автоэнкодеры** (англ. autoencoders). Автоэнкодеры – это нейронные сети, которые обучаются на входных данных и затем используются для восстановления этих данных. Если входные данные содержат аномалии, то автоэнкодеры могут обнаружить эти аномалии и выдать предупреждение [7].

Также для решения задачи анализа HTTP-трафика на предмет аномалий и атак можно использовать **сверточные нейронные сети** (англ. convolutional neural networks), которые могут анализировать временные ряды данных и выделять из них признаки, связанные с аномалиями [8].

Рекуррентные нейронные сети (англ. recurrent neural networks) также могут использоваться для анализа HTTP-трафика, особенно если данные имеют последовательную структуру.

Выбор конкретного типа нейронной сети зависит от характеристик данных и требований к точности и скорости обнаружения ано-

малий [9]. Для решения задачи было решено выбрать рекуррентные нейронные сети.

Так как длина HTTP-пакетов может быть довольно большой, а векторное представление слов для анализа не является возможным, в виду отсутствия семантического смысла, а также наличия различных типов данных помимо текстового, необходим тип рекуррентных нейронных сетей, способный 2обрабатывать длинную последовательность данных.

Для решения задачи было решено использовать рекуррентные сети типа LSTM и GRU, а после сравнения эффективности выбрать окончательный вариант.

Для перехвата пакетов и их дальнейшего анализа используется open-source инструмент mitmproxy. В данный инструмент входит функциональность обратного прокси-сервера, который, в отличие от обычного, работает на стороне сервера, а не клиента, что позволяет поддерживать HTTP-соединение со многими клиентами, при этом перехватывая и/или обрабатывая трафик.

Основной модуль перехватывает пакеты, а также производит первичную обработку (проверка на количество переноса строк в запросе).

Модуль нейросети использует методы для получения массивов данных из файлов с помощью библиотеки pickle, затем преобразование этих данных в массивы numpy и сохранение в файлы pickle (для оптимизации тестирования и работы). После этого происходит преобразование данных в нужный вид для обработки нейросетью. После чего создается непосредственно модель нейросети, ее обучение и тестирование с помощью keras – открытой библиотеки Python для работы с различными типа нейросетей. Фрагмент кода представлен на рис. 2.

```
# соединяем массивы
signs = np.concatenate((data_anomal1_sign, data_normal1_sign))

# вертикально соединяем массивы
data = np.vstack((data_anomal1, data_normal1))

# X - массив признаков, y - массив меток классов
X_train, X_test, y_train, y_test = train_test_split(data, signs, test_size=0.2, random_state=42)


# Создание модели
model = Sequential()
model.add(Dense(64, input_dim=X_train.shape[1], activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])

# Обучение модели
model.fit(X_train, y_train, epochs=100, batch_size=32, validation_data=(X_test, y_test))

# Оценка качества модели
score = model.evaluate(X_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

Рис. 2. Фрагмент кода модуля нейросети.

В качестве тестирования была взята простая рекуррентная нейронная сеть. После обучения, коэффициент точности при тестировании составил 95%.

Заключение

Был разработан модуль киберполигона для построения виртуальных сетей, реализующий следующие функции:

• Поддержка создания пользовательских виртуальных машин в среде построения лаборатории.

• Создание виртуальных сетевых интерфейсов у машин.

• Создание изолированных виртуальных сетей для объединения виртуальных интерфейсов машин.

• Удаленное подключение к виртуальным машинам.

• Настройка виртуальных машин (ограничение используемой оперативной памяти и процессорного времени).

• Сохранение конфигураций лабораторий, виртуальных сетей и машин.

• Наличие графического интерфейса для управления лабораторией.

Для модуля защиты веб-сервисов на ос-

нове нейросети в результате анализа были выбраны нейросети типа LSTM и GRU. Также были выбраны инструменты для реализации выдвинутых требований к модулю, созданы основные модули программы, а также создана тестовая нейросеть. На основании полученных результатов можно сделать вывод, что разработка в данном направлении может открыть новые возможности для специалистов в области информационной безопасности.

Литература

1. Рахметов Р. Роль киберполигона в обеспечении ИБ / Рахметов Р. [Электронный ресурс] // Security Vision: [сайт]. – URL: https://www.securityvision.ru/blog/rol-kiberpoligona-v-obespechenii-ib/# (дата обращения 10.02.2024г.)

2. Сарычев Д. Как выбрать Web Application Firewall в 2023 году / Сарычев Д. [Электронный ресурс] // Anti-Malware [сайт]. – URL: https://www.anti-malware.ru/analytics/Technology_Analysis/Web-Application-Firewall-in-2023#part34 (дата обращения: 11.02.2024г.)

3. Scott H. Software Containers: Used More Frequently than Most Realize / Scott H. [Электронный ресурс] // Network World [сайт]. – URL: https://www.networkworld.com/article/749098/cisco-subnet-software-containers-used-more-frequently-than-most-realize.html (дата обращения 10.02.2024г.)

4. Berl Andreas, Fischer Andreas, Hermann de Meer. Using System Virtualization to Create Virtualized Networks. — Berlin: Electronic Communications of the EASST, 2015. — 12 p.

5. Matthew Portnoy. Virtualization Essentials. – New York: Sybex, 2023. – 196 p.

6. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning – Cambridge: MIT Press, 2016. – 555 p.

7. Питер Яворски. Ловушка для багов: полевое руководство по веб-хакингу. – СПб.: Питер, 2020. – 28 с.

8. Anthony Williams. Convolutional Neural Networks in Python: Introduction to Convolutional Neural Networks. – Scotts Valley: Createspace, 2019. – 57 p.

9. Aurélien Géron. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. – North Carolina: O'Reilly Media, 2019. – 152 p.

References

1. Rakhmetov R. Rol' kiberpoligona v obespechenii IB / Rakhmetov R. [Elektronnyy resurs] // Security Vision: [sayt]. – URL: https://www.securityvision.ru/blog/rol-kiberpoligona-v-obespechenii-ib/# (data obrashcheniya 10.02.2024g.)

2. Sarychev D. Kak vybrat' Web Application Firewall v 2023 godu / Sarychev D. [Elektronnyy resurs] // Anti-Malware [sayt]. – URL: https://www.anti-malware.ru/analytics/Technology_Analysis/Web-Application-Firewall-in-2023#part34 (data obrashcheniya: 11.02.2024g.)

3. Scott H. Software Containers: Used More Frequently than Most Realize / Scott H. [Electronic resource] // Network World [website]. – URL: https://www.networkworld.com/article/749098/cisco-subnet-software-containers-used-more-frequently-than-most-realize.html (дата обращения 10.02.2024г.)

4. Berl Andreas, Fischer Andreas, Hermann de Meer. Using System Virtualization to Create Virtualized Networks. — Berlin: Electronic Communications of the EASST, 2015. — 12 p.

5. Matthew Portnoy. Virtualization Essentials. – New York: Sybex, 2023. – 196 p.

6. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning – Cambridge: MIT Press, 2016. – 555 p.

7. Piter Yavorski. Lovushka dlya bagov: polevoe rukovodstvo po veb-khakingu. – SPb.: Piter, 2020. – 28 s.

8. Anthony Williams. Convolutional Neural Networks in Python: Introduction to Convolutional Neural Networks. – Scotts Valley: Createspace, 2019. – 57 p.

9. Aurélien Géron. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. – North Carolina: O'Reilly Media, 2019. – 152 p.

КУЗЬМИНА Ульяна Владимировна, кандидат технических наук, доцент по специальности «Информационная безопасность автоматизированных систем», доцент кафедры информатики и информационной безопасности федерального государственного бюджетного образовательного учреждения высшего образования «Магнитогорский государственный техниче-

ский университет им. Г.И. Носова». 455000, г. Магнитогорск, пр. Ленина, 38. E-mail: u.mihaylova@magtu.ru

KUZMINA Ulyana Vladimirovna, Candidate of Technical Sciences, Associate Professor in the specialty «Information Security of automated systems», Associate Professor of the Department of Computer Science and Information Security of the Federal State Budgetary Educational Institution of Higher Education «Nosov Magnitogorsk State Technical University». 455000, Magnitogorsk, Lenin Ave., 38. E-mail: u.mihaylova@magtu.ru

МИХАЙЛОВА Ольга Евгеньевна, студент кафедры информатики и информационной безопасности федерального государственного бюджетного образовательного учреждения высшего образования «Магнитогорский государственный технический университет им. Г.И. Носова». 455000, г. Магнитогорск, пр. Ленина, 38. E-mail: olgamihailova01@mail.com

MIKHAYLOVA Olga Evgenevna, student of the department of computer science and information security, Federal State Budgetary Educational Institution of Higher Education «Nosov Magnitogorsk State Technical University». 455000, Magnitogorsk, Lenin Ave., 38. E-mail: olgamihailova01@mail.com

АФАНАСЬЕВ Юрий Петрович, студент кафедры информатики и информационной безопасности федерального государственного бюджетного образовательного учреждения высшего образования «Магнитогорский государственный технический университет им. Г.И. Носова». 455000, г. Магнитогорск, пр. Ленина, 38. E-mail: yurashca19@mail.ru

AFANASYEV Yuri Petrovich, student of the department of computer science and information security, Federal State Budgetary Educational Institution of Higher Education «Nosov Magnitogorsk State Technical University». 455000, Magnitogorsk, Lenin Ave., 38. E-mail: yurashca19@mail.ru