Обнаружение SSRF-уязвимостей с помощью методов обучения с учителем

Г. Г. Чавес Кирос
Санкт-Петербургский политехнический университет Петра Великого
chaveskiros.g@edu.spbstu.ru

Н. В. Воинов
Санкт-Петербургский политехнический университет Петра Великого
voinov@ics2.ecd.spbstu.ru

И. В. Зайцев
Санкт-Петербургский политехнический университет Петра Великого
zajtsev_iv@spbstu.ru

П. Д. Дробинцев
Санкт-Петербургский политехнический университет Петра Великого
drob@ics2.ecd.spbstu.ru

Аннотация. **Работа посвящена применению методов машинного обучения с учителем для обнаружения подделки запросов на стороне сервера (Server-Side Request Forgery, SSRF) – определенного типа уязвимости в веб-приложениях, при которой злоумышленник может отправлять фальшивые запросы с сервера на другие внутренние или внешние ресурсы. Проведено обучение и тестирование различных моделей с целью выявления SSRF-угроз. Полученные результаты применения обученных моделей доказывают их высокую эффективность в обнаружении уязвимостей данного типа.**

Ключевые слова: подделка запросов на стороне сервера (SSRF), обучение с учителем, обнаружение уязвимостей, безопасность веб-приложений, классификация

I. Введение

При повсеместном распространении интернета веб-приложения стали неотъемлемой частью современной жизни, в том числе в коммерческой сфере. От платформ электронной коммерции до услуг онлайн-банкинга эти приложения позволяют пользователям быстро получать доступ к информации и выполнять транзакции из любой точки мира. Однако это удобство также влечет за собой значительные проблемы безопасности, поскольку веб-приложения часто становятся целями кибератак, использующих уязвимости в дизайне, реализации или конфигурации. Обеспечение безопасности этих приложений имеет огромное значение, поскольку нарушения могут привести к потере конфиденциальных данных и подорвать доверие пользователей [1].

Обнаружение уязвимостей помогает своевременно предпринимать усилия по их устранению и играет ключевую роль в обеспечении безопасности веб-приложений. Согласно Ресурсному центру компьютерной безопасности (Computer Security Resource Center CSRC), уязвимость – это «слабое место в информационной системе, процедурах безопасности системы, внутреннем контроле или реализации, которое может быть использовано или вызвано источником угрозы». [2]. Раннее выявление таких слабых мест имеет решающее значение для предотвращения атак, которые могут поставить под угрозу целые системы или нарушить работу служб. Для этого широко используются традиционные методы, такие как ручные проверки кода, тестирование на возможность проникновения и автоматизированные инструменты на основе правил. Однако эти подходы плохо адаптированы к сложным приложениям, требуют много ресурсов и часто приводят к высоким показателям ложных срабатываний и отрицательных результатов, особенно в динамических средах, таких как PHP [3].

Подделка запросов на стороне сервера (англ., Server-Side Request Forgery, SSRF) занимает 10-е место среди самых критичных уязвимостей в рейтинге OWASP Top Ten 2021 [4]. SSRF-уязвимости возникают, когда веб-приложение извлекает удаленный ресурс без надлежащей проверки предоставленных пользователем URL-адресов, что позволяет злоумышленникам манипулировать запросами на стороне сервера и получать доступ к внутренним системам или службам. Этот тип уязвимости особенно опасен в облачных средах, где внутренние службы более доступны, и злоумышленники могут использовать неверные конфигурации для получения несанкционированного доступа [5]. Последствия SSRF-атак могут быть очень серьезными: от несанкционированного доступа к данным и кражи учетных данных до компрометации внутренней инфраструктуры. Растущая сложность архитектур программного обеспечения в сочетании с широким распространением облачных служб еще больше усилили риски, связанные с SSRF, что делает эти уязвимости насущной проблемой для организаций по всему миру.

Для устранения недостатков традиционных методов обнаружении уязвимостей веб-приложений все чаще применяют искусственный интеллект (ИИ) и машинное обучение (МО). Подходы на основе МО показали перспективные результаты в выявлении различных пробелов безопасности путем анализа больших наборов данных и распознавания сложных шаблонов. Однако их

применение для обнаружения SSRF остается малоизученным, необходимы дальнейшие исследования для оценки их эффективности при этом конкретном типе уязвимостей. Используя ключевые особенности веб-трафика, такие как URL-структуры, HTTP-заголовки и поведение запросов, методы МО могут обеспечить масштабируемое и адаптивное решение для обнаружения SSRF.

В статье предлагается подход, основанный на применении методов обучения с учителем, для автоматизированного обнаружения SSRF-уязвимостей с использованием набора данных, сгенерированного на основе различных сценариев SSRF. В ходе работы были обучены и протестированы различные модели для оценки их эффективности в выявлении данных уязвимостей.

II. Обзор литературы

Использование методов МО для автоматизированного обнаружения уязвимостей в веб-приложениях представлено во многих работах. Например, в работе, рассматривающей обучение с подкреплением [6], предложена модель на основе Q-обучения, реализованная с помощью TensorFlow для обнаружения уязвимостей в реальном времени, демонстрируя высокую эффективность по сравнению с традиционными методами. В другом исследовании описан метод на основе МО для обнаружения уязвимостей межсайтового скриптинга (Cross-Site Scripting, XSS) с помощью функции идентификации кода проверки с низким уровнем ложных срабатываний [7]. В работе [8] представлено решение на основе обучения с учителем с целью обнаружения уязвимостей подделки межсайтовых запросов (Cross-site Request Forgery, CSRF) методом черного ящика, в рамках работы было проведено обучение на 5828 HTTP-запросах, в результате удалось успешно идентифицировать новые уязвимости на популярных веб-сайтах. В других работах применялся алгоритм случайного леса (Random Forest) для обнаружения уязвимостей веб-приложений путем извлечения признаков из URL-адресов [9] и, объединяя статический анализ и методы МО, для выявления уязвимостей в приложениях PHP с использованием собственных токенов и абстрактных синтаксических деревьев (AST), достигая показателя полноты (recall) в 92 % [10]. Еще одно исследование, посвященное обнаружению SQL-инъекций с использованием нескольких алгоритмов машинного обучения (наивный байесовский классификатор, метод опорных векторов (SVM), дерево решений, случайный лес, XGBoost и CatBoost) продемонстрировало показатели точности, полноты и F1-меры (F1-Score) выше 95 % [11].

Некоторые работы посвящены и SSRF-уязвимостям. В [12] анализируются SSRF-уязвимости в приложениях PHP посредством статического анализа потока данных с использованием графов свойств кода (Code Property Graphs, CPG) для выявления опасных потоков. В [13] представлен SSRFuzz - автоматизированный инструмент для обнаружения SSRF-уязвимостей в приложениях PHP, который был протестирован на 27 реальных приложениях, обнаружив 28 уязвимостей, 16 из которых входят в базу данных общеизвестных уязвимостей информационной безопасности (Common Vulnerabilities and Exposures, CVE).

Рассмотренные исследования показывают, что методы МО помогают эффективно обнаруживать такие уязвимости веб-приложений, как SQL-инъекции, XSS и CSRF. В свою очередь, подходы, разработанные для обнаружения SSRF, включают в основном статический анализ или автоматизированные инструменты, такие как SSRFuzz. Таким образом, исследование применения методов обучения с учителем для автоматизированного обнаружения SSRF-уязвимостей является актуальной задачей.

III. Предлагаемый подход

Предлагаемый подход к обнаружению SSRF-уязвимостей с использованием МО основан на традиционной схеме обучения с учителем, начиная с генерации набора данных до оценки результатов работы модели. Это стандартная схема задачи классификации в данном случае направлена на анализ сгенерированного веб-трафика для выявления аномальных шаблонов, которые могут указывать на наличие SSRF-уязвимостей. Данный подход не учитывает какие-либо предопределенные правила или анализ исходного кода, что делает его применимым для различных SSRF-сценариев. Основные этапы подхода описаны ниже.

A. Генерация данных

На первом этапе был сгенерирован синтетический набор данных, содержащий 15 классов SSRF-уязвимостей, включая примеры легитимного трафика и различные SSRF-сценарии. Данные были сбалансированы для равного количества образцов для каждого класса.

B. Предобработка данных

HTTP-запросы были структурированы путем извлечения ключевых признаков, таких как URL, HTTP-параметры и заголовки запросов, которые являются наиболее важными для обнаружения SSRF. Также для повышения производительности моделей МО была выполнена нормализация данных.

C. Выбор и обучение моделей

Были выбраны несколько алгоритмов обучения с учителем, которые признаны эффективными в задачах классификации сложных шаблонов веб-трафика: Random Forest, XGBoost, LightGBM, Logistic Regression, Decision Trees, Extra Trees и Ensembles (Voting and Stacking Classifier). Для генерализации моделей была выполнена кросс-валидация и настройка гиперпараметров. После этого проведена балансировка классов с интеграцией реальных данных из общедоступных наборов данных и реального трафика.

D. Оценка моделей

Модели оцениваются с использованием стандартных метрик производительности, таких как точность, полнота, F1-мера и ROC-AUC, для измерения эффективности обнаружения SSRF-уязвимостей. На этом этапе сгенерированные данные используются для проверки способности моделей выявлять уязвимости в различных

сценариях, учитывая как вредоносный, так и смоделированный легитимный трафик.

В рамках работы были оценены несколько алгоритмов МО, описанных ниже.

Random Forest – это алгоритм обучения с учителем, основанный на ансамблевых деревьях решений. Он объединяет несколько деревьев, обученных на случайных подмножествах данных и признаков, что снижает риск переобучения и улучшает обобщение. Каждое дерево делает прогноз, а окончательный результат определяется голосованием большинства (классификация) или усреднением (регрессия).

XGBoost – это алгоритм, который итеративно улучшает последовательные модели. Он использует низкопроизводительные деревья решений, которые корректируются на каждой итерации, что позволяет оптимизировать как точность, так и скорость обработки. Он очень эффективен для больших объемов данных.

LightGBM – это оптимизированная версия градиентного бустинга, разработанная для работы с большими наборами данных. Он использует подход снижения вычислительной сложности, что делает его особенно быстрым и подходящим для задач классификации в больших объемах данных, таких как веб-трафик для обнаружения SSRF-уязвимостей.

Extra Trees – это алгоритм, который, как и Random Forest, использует деревья решений, но с большей случайностью в разбиениях узлов. Такой подход улучшает обобщение и предсказательную силу модели, делая ее быстрым и эффективным алгоритмом классификации закономерностей в сложных данных.

IV. Результаты

Для оценки эффективности выбранных моделей в обнаружении SSRF-уязвимостей была проведена серия экспериментов с использованием набора данных, состоящего из 15000 образцов, равномерно распределенных по 15 классам, представляющим различные SSRF-сценарии наряду с легитимным трафиком. Перед обучением данные прошли предварительную обработку, которая включала извлечение ключевых признаков из HTTP-запросов с последующей нормализацией и кодированием категориальных переменных для повышения производительности модели. Затем набор данных был разделен на обучающий (70 %) и тестовый (30 %) наборы для обеспечения надежной оценки модели.

Для классификации использовались различные алгоритмы обучения с учителем, включая древовидные модели, линейные классификаторы и ансамблевые методы. Модели обучались с использованием перекрестной проверки, а для оптимизации их производительности выполнялась настройка гиперпараметров. Для оценки способности каждой модели различать легитимные запросы и различные SSRF-шаблоны использовались метрики точности, полноты, F1-меры и ROC-AUC.

Результаты, представленные в табл. I, демонстрируют эффективность классификации. Модели на основе деревьев, включая Random Forest, XGBoost, LightGBM и Extra Trees, достигают наивысших показателей точности, полноты и F1-меры в 96 %. Кроме того, их оценка ROC-AUC близка к идеальной (99,9 %), демонстрируя отличную способность различать SSRF-уязвимости и легитимный трафик. Логистическая регрессия, хотя и немного менее эффективна (точность 92,3 %), остается хорошим вариантом из-за своей простоты и интерпретируемости. Модель дерева решений с точностью 94,3 % показывает достойную производительность, но ее более низкая оценка ROC-AUC (97,3 %) говорит о том, что она менее эффективна при разделении классов по сравнению с моделями на основе ансамблей.

ТАБЛИЦА I. Результаты работы моделей по обнаружению SSRF-уязвимостей (%)

Модель	**Точность (Accuracy)**	**Точность (Precision)**	**Полнота (Recall)**	**F1-Score**	**ROC-AUC**
Random Forest	96	96,4	96	96	99,9
XGBoost	96	96,2	96	96	99,9
LightGBM	96	96	96	96	99,9
Logistic Regression	92,3	93,2	92,3	92,1	99,8
Decision Tree	94,3	94,7	94,3	94,3	97,3
Extra Trees	95,7	96	95,7	95,7	99,9
Ensemble (Voting)	96	96	96	96	99,9
Ensemble (Stacking)	96,3	96,4	96,3	96,3	99,9

Методы ансамблей (Voting and Stacking Classifier), дали результаты, сопоставимые с лучшими отдельными моделями, что указывает на то, что объединение моделей не обеспечило значительного повышения производительности.

V. Заключение

Статья посвящена применению и оценке моделей МО при обнаружении уязвимостей SSRF-уязвимостей, которые остаются в значительной степени неисследованными в контексте автоматизированного обнаружения. Результаты показывают, что такие модели, как Random Forest, XGBoost, LightGBM и Extra Trees, достигают высокой точности (96 %) и почти идеального показателя ROC-AUC (99,9 %), что указывает на их отличную способность различать SSRF-уязвимости и легитимный трафик.

Несмотря на эти многообещающие результаты, использование МО для обнаружения веб-уязвимостей, в частности SSRF, все еще находится на ранних стадиях. По сравнению с другими угрозами безопасности, SSRF-уязвимости получили ограниченное внимание в исследованиях на основе МО, что подчеркивает необходимость дальнейшего изучения. Данная статья дает основу для разработки более совершенных механизмов обнаружения, которые используют МО для повышения веб-безопасности.

В рамках следующего этапа исследования планируется интегрировать реальные наборы данных для оценки производительности моделей. Кроме того,

необходимо изучить различные методы выбора признаков и альтернативные представления данных для потенциального улучшения возможностей обнаружения. Также необходимо изучить подходы глубокого обучения для улучшения идентификации сложных SSRF-шаблонов.

Список литературы

[1] Yadav N.S., Rounak R., Sharma P.C. Web-based Vulnerability Analysis and Detection. International Journal of Sensors, Wireless Communications and Control. 2024, vol. 15. DOI: 10.2174/0122103279319619241008221647

[2] *Computer Security Resource Center (CSRC), Vulnerability definition*. Available at: https://csrc.nist.gov/glossary/term/vulnerability (accessed 12 March 2025)

[3] Ji Y., Dai T., Tang Y., He J. Poster: Whether We Are Good Enough to Detect Server-Side Request Forgeries in PHP-native Applications? Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security. 2024, pp. 4928-4930. DOI: 10.1145/3658644.3691419

[4] *OWASP Top Ten 2021*. Available at: https://owasp.org/Top10/ (accessed 12 March 2025)

[5] Jabiyev B., Mirzaei O., Kharraz A., Kirda E. Preventing server-side request forgery attacks. in Proceedings of the 36th Annual ACM Symposium on Applied Computing. 2021, pp. 1626–1635. DOI: 10.1145/3412841.3442036

[6] Vybornova O.N., Ryzhikov A.N. Automated search for web application vulnerabilities based on reinforcement learning. Caspian journal: control and high technologies. 2021, vol. 53, pp. 91-97.

[7] Hu L., Chang J., Chen Z., Hou B. Web application vulnerability detection method based on machine learning. Journal of Physics: Conference Series. 2021, vol. 1827, no. 1, p. 012061. DOI: 10.1088/1742-6596/1827/1/012061

[8] Calzavara S., Conti M., Focardi R., Rabitti F., Tolomei G. Mitch: A Machine Learning Approach to the Black-Box Detection of CSRF Vulnerabilities. 2019 IEEE European Symposium on Security and Privacy (EuroS&P). 2019, pp. 528-543. DOI: 10.1109/EuroSP.2019.00045

[9] Singh C., Vijayalakshmi V., Raj H. A Machine Learning Approach for Web Application Vulnerability Detection Using Random Forest. International Journal for Research in Applied Science & Engineering Technology. 2022, vol. 10, no. 12, pp. 2106-2112. DOI: 10.22214/ijraset.2022.48397

[10] Anbiya D.R., Purwarianti A., Asnar Y. Vulnerability detection in php web application using lexical analysis approach with machine learning. 2018 5th International Conference on Data and Software Engineering (ICoDSE). 2018, pp. 1-6.

[11] Zhumabekova A., Matson E.T., Karyukin V., Zhumabekova K., Zhuandykov B., Ussatova O., Telbayeva T. Determining Web Application Vulnerabilities Using Machine Learning Methods. 2023 19th International Asian School-Seminar on Optimization Problems of Complex Systems (OPCS). 2023, pp. 136-139. DOI: 10.1109/ICODSE.2018.8705809

[12] Wessels M., Koch S., Pellegrino G., Johns M. SSRF vs. Developers: A Study of SSRF-Defenses in PHP Applications. 33rd USENIX Security Symposium (USENIX Security 24). 2024, pp. 6777-6794.

[13] Wang E., Chen J., Xie W., Wang C., Gao Y., Wang Z., Duan H., Liu Y., Wang B. Where URLs become weapons: Automated discovery of SSRF vulnerabilities in web applications. 2024 IEEE Symposium on Security and Privacy (SP). 2024, pp. 239-257. DOI: 10.1109/SP54263.2024.00198