Russian Technological Journal.
2023;11(4):7–15

ISSN 2782-3210 (Print)
ISSN 2500-316X (Online)

УДК 004.89
https://doi.org/10.32362/2500-316X-2023-11-4-7-15

НАУЧНАЯ СТАТЬЯ

Построение верификатора стойкости пароля с использованием классических методов машинного обучения и рекуррентной LSTM нейронной сети

В.В. Беликов [1, @],
И.А. Прокуронов [2]

[1] *МИРЭА – Российский технологический университет, Москва, 119454 Россия*
[2] *СФБ Лаборатория, Москва, 127083 Россия*
[@] *Автор для переписки, e-mail: belikov_v@mirea.ru*

Резюме

Цель. Аутентификация с использованием паролей является одним из наиболее распространенных способов проверки подлинности в компьютерных системах. Существующие атаки на пароли, включающие в себя, в т.ч. атаки перебора и атаки по словарю, требуют не только защиты учетных данных пользователя на этапе эксплуатации паролей, но и определения требований к паролю, позволяющих повысить стойкость пароля к атакам, минимизируя возможность их реализации злоумышленником. Важной задачей при этом становится разработка верификатора, осуществляющего проверку пароля на стойкость и позволяющего исключить задание пользователем паролей, подверженных взлому. Построение верификатора с использованием методов машинного обучения позволяет алгоритмам самим формулировать требования к сложности пароля в произвольно комплексной форме, отталкиваясь только от инцидентов, имеющихся для каждой категории стойкости списков известных паролей.

Методы. Предложены алгоритмы машинного обучения с учителем: метод опорных векторов, случайный лес, бустинг, рекуррентная LSTM (long short-term memory) нейронная сеть. В эксперименте для предобработки данных применены метод простой индексации символов с последующей обработкой embedding-слоем и метод TF-IDF (term frequency-inverse document frequency). Для выбора гиперпараметров алгоритмов была использована кросс-валидация.

Результаты. Проведен анализ рекомендаций и требований к паролям в международных и отечественных стандартах и возможности их реализации в виде верификатора стойкости пароля в различных операционных системах. Приведены результаты эксперимента на существующем наборе помеченных по уровню стойкости паролей. Проведена их оценка с использованием macro f1-меры.

Выводы. Использование рекуррентной LSTM нейронной сети выделено как одно из наиболее перспективных направлений для построения верификатора стойкости пароля.

Ключевые слова: компьютерная безопасность, стойкость пароля, машинное обучение с учителем, рекуррентная нейронная сеть, LSTM

• **Поступила:** 11.12.2022 • **Доработана:** 01.02.2023 • **Принята к опубликованию:** 02.05.2023

Для цитирования: Беликов В.В., Прокуронов И.А. Построение верификатора стойкости пароля с использованием классических методов машинного обучения и рекуррентной LSTM нейронной сети. *Russ. Technol. J.* 2023;11(4):7–15. https://doi.org/10.32362/2500-316X-2023-11-4-7-15

Прозрачность финансовой деятельности: Авторы не имеют финансовой заинтересованности в представленных материалах или методах.

Авторы заявляют об отсутствии конфликта интересов.

RESEARCH ARTICLE

Password strength verification based on machine learning algorithms and LSTM recurrent neural networks

Vladimir V. Belikov [1, @], Ivan A. Prokuronov [2]

[1] *MIREA – Russian Technological University, Moscow, 119454 Russia*
[2] *SFB Laboratory, Moscow, 127083 Russia*
[@] *Corresponding author, e-mail: belikov_v@mirea.ru*

Abstract

Objectives. One of the most commonly used authentication methods in computer systems, password authentication is susceptible to various attacks including brute-force and dictionary attacks. This susceptibility requires not only the strict protection of user credentials, but also the definition of criteria for increasing a password's strength to minimize the possibility of its exploitation by an attacker. Thus, an important task is the development of a verifier for checking passwords for strength and prohibiting the user from setting passwords that are susceptible to cracking. The use of machine learning methods to construct a verifier involves algorithms for formulating requirements for password complexity based on lists of known passwords available for each strength category.

Methods. The proposed supervised machine learning algorithms comprise support vector machines, random forest, boosting, and long short-term memory (LSTM) recurrent neural network types. Embedding and term frequency–inverse document frequency (TF-IDF) methods are used for data preprocessing, while cross-validation is used for selecting hyperparameters.

Results. Password strength recommendations and requirements from international and Russian standards are described. The existing methods of password strength verification in various operating systems are analyzed. The experimental results based on existing datasets comprising passwords having an associated level of strength are presented.

Conclusions. A LSTM recurrent neural network is highlighted as one of the most promising areas for building a password strength verifier.

Keywords: cybersecurity, password strength, supervised machine learning, recurrent neural network, LSTM

• **Submitted:** 11.12.2022 • **Revised:** 01.02.2023 • **Accepted:** 02.05.2023

For citation: Belikov V.V., Prokuronov I.A. Password strength verification based on machine learning algorithms and LSTM recurrent neural networks. *Russ. Technol. J.* 2023;11(4):7–15. https://doi.org/10.32362/2500-316X-2023-11-4-7-15

Financial disclosure: The authors have no a financial or property interest in any material or method mentioned.

The authors declare no conflicts of interest.

ВВЕДЕНИЕ

Аутентификация с использованием паролей является одним из наиболее распространенных способов проверки подлинности в компьютерных системах [1]. Существующие атаки на пароли (в т.ч. атаки перебора, атаки по словарю и атаки с помощью радужных таблиц) требуют не только защиты учетных данных пользователя на этапе эксплуатации паролей, но и определения требований к паролю при его задании. Эти требования должны повышать стойкость пароля к указанным атакам, минимизируя возможность их реализации злоумышленником [2]. Важной задачей при этом становится разработка верификатора, осуществляющего проверку пароля на стойкость и позволяющего исключить задание пользователем паролей, подверженных взлому [3].

АНАЛИЗ СУЩЕСТВУЮЩИХ ПОДХОДОВ К ПОСТРОЕНИЮ ВЕРИФИКАТОРА СТОЙКОСТИ ПАРОЛЯ

Наиболее известные правила задания стойкого пароля представлены [4] в следующих документах:

1. Специальная публикация Национального института стандартов и технологий (NIST) «Рекомендации по цифровой идентификации. Аутентификация и управление жизненным циклом»[1].
2. Методический документ Федеральной службы по техническому и экспортному контролю (ФСТЭК) «Меры защиты информации в государственных информационных системах»[2].

Эти руководящие принципы содержат рекомендации для пользователей при создании паролей, а также требования и рекомендации для верификаторов (веб-сайты, программное обеспечение и т.д.), которые содержат в себе систему проверки и обработки паролей.

В первом документе пароли фигурируют под названием «запомненные секреты» (от англ. memorized secrets). Среди основных положений можно выделить следующие:

1. Запомненные секреты (пароли) должны быть длиной не менее 8 символов, если они были выбраны пользователем.
2. Запомненные секреты (пароли), случайно сгенерированные компьютером или верификатором, должны иметь длину не менее 6 символов, а также могут состоять полностью из цифр.
3. Если компьютер или верификатор запрещает выбранный сохраненный секрет (пароль) на основании того, что он содержится в принятом ранее черном списке скомпрометированных значений, пользователь должен выбрать другой сохраненный секрет (пароль).

В документе ФСТЭК рекомендованная минимальная длина пароля составляет 6 символов для достижения требований четвертого (самого простого) уровня защищенности персональных данных [5].

При обработке запросов на установление и изменение сохраненных секретов верификаторы должны сравнивать предполагаемые секреты со списком часто используемых, ожидаемых или скомпрометированных паролей. Например, список может включать в себя:

1. Пароли, полученные из баз взломанных паролей.
2. Словарные слова.
3. Повторяющиеся или последовательные символы (например, «qqqqqq», «qwerty12345»).
4. Слова, зависящие от контекста, такие как название службы, имя пользователя и их производные (например, «mireastudent», «ivanivanov»).

Еще одним способом проверки пароля на его стойкость является использование специальных сервисов, таких как, например, сервис, предложенный на официальном сайте компании Kaspersky[3]. Этот сервис позволяет получить информацию о стойкости пароля, о встречаемости пароля в утекших базах, а также узнать сколько потребуется времени, чтобы взломать пароль с помощью атаки грубой силы (рис. 1).

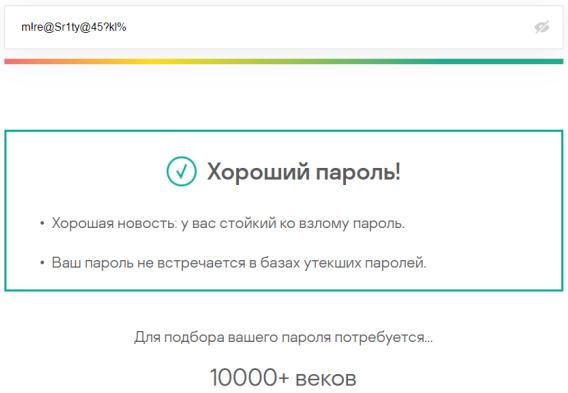

Рис. 1. Демонстрация проверки пароля на стойкость на сайте Kaspersky

[1] https://pages.nist.gov/800-63-3/sp800-63b.html. Дата обращения 01.02.2023. / Accessed February 01, 2023.

[2] https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/metodicheskij-dokument-ot-11-fevralya-2014-g (in Russ.). Дата обращения 01.02.2023. / Accessed February 01, 2023.

[3] https://password.kaspersky.com/ru/ (in Russ.). Дата обращения 01.02.2023. / Accessed February 01, 2023.

Сложность выбираемых пользователем паролей может быть охарактеризована с использованием концепции теории информации, основоположником которой считается американский математик Клод Шеннон. Хотя энтропию, как меру информационной емкости системы, можно легко вычислить для данных, имеющих детерминированные функции распределения, оценка энтропии для паролей, выбранных пользователем, является сложной задачей, и предыдущие попытки сделать это не привели к получению точного результата. По этой причине, например, в публикации NIST используется подход, основанный в первую очередь на длине пароля.

Для обеспечения проверки пароля на стойкость в различных операционных системах существуют специальные модули, которые оценивают пароль по каким-либо критериям, придерживаясь общепринятых рекомендаций, а также рекомендаций, которые могут быть установлены администратором системы. В общем случае это называется политикой паролей, которая представляет собой некий набор правил, повышающий безопасность учетных записей пользователей, поощряя их к использованию более надежных паролей.

Долгое время в операционных системах Linux использовался PAM-модуль (PAM – подключаемые модули аутентификации, pluggable authentication modules) *pam_cracklib*, который предназначался для проверки пароля по словарным словам [6]. В последних версиях Linux данный модуль был заменен на модуль *pam_pwquality*, основанный на модуле *pam_cracklib* и полностью обратно совместимый с его опциями. Этот модуль делает подход к созданию политик более простых паролей для проверки того, что пользователи принимают рекомендации и требования администратора по созданию пароля.

Примером политики паролей, которую можно задать с использованием указанного модуля, может быть следующий список требований:

1. Минимальная длина пароля – 10 символов.
2. В новом пароле должно присутствовать 6 новых символов, которых не было в старом пароле.
3. Пароль должен содержать строчные буквы, заглавные буквы, специальные символы, а также цифры.
4. В пароле не должно содержаться более двух подряд повторяющихся символов.
5. В пароле не более шести подряд идущих символов должны быть из одного класса.
6. Присутствие проверки GESOC.
7. Запретить слова «mirea», «security», «admin», «password», «cyber».

Содержание файла конфигурации модуля */etc/security/pwquality.conf*, отражающего описанную политику безопасности, приведено на рис. 2.

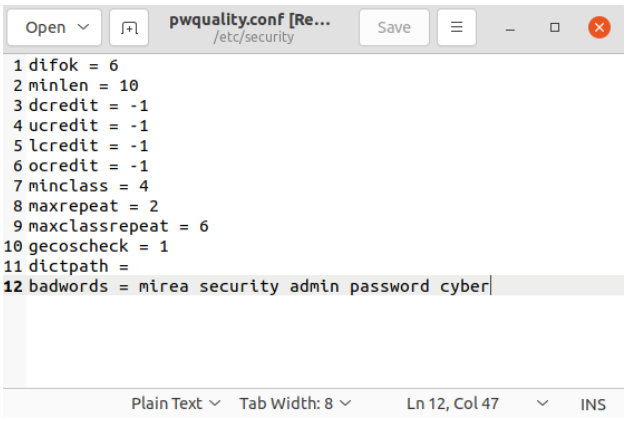

Рис. 2. Вид файла конфигурации *pwquality.conf*

Для проверки корректности заданной политики паролей можно воспользоваться программой *pwscore*, которая входит в пакет *libpwquality*. Пример тестирования пароля, содержащего запрещенное слово, и пароля, удовлетворяющего всем требованиям, с использованием утилиты *pwscore*, выставляющей баллы от 0 до 100, приведен на рис. 3.

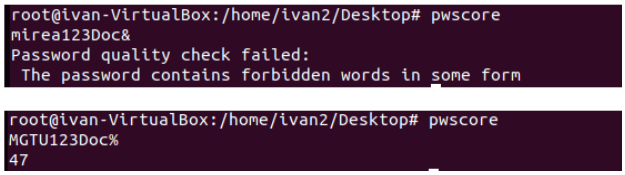

Рис. 3. Тестирование пароля

В операционных системах семейства Windows настройка политики паролей задается в разделе «Локальная политика безопасности» утилиты *gpedit.msc* [7] (рис. 4).

Имея в своем арсенале всего лишь восемь настраиваемых параметров, операционная система Windows позволяет строго устанавливать и менять политику паролей, включая или включая и комбинируя установленные значения этих параметров. Одним из самых важных при этом является пункт «Пароль должен отвечать требованиям сложности». Включение данного пункта определяет то, что задаваемые пароли должны удовлетворять следующим требованиям:

1. Пароль не должен содержать в себе имя учетной записи пользователя или частей полного имени пользователя длиной более двух подряд идущих символов.
2. Минимальная длина пароля 6 символов.
3. Пароль должен содержать символы, как минимум, из трех классов: латинские заглавные буквы, латинские строчные буквы, цифры, специальные символы. В Windows 11 был добавлен четвертый класс, который требует наличие любого символа Unicode.

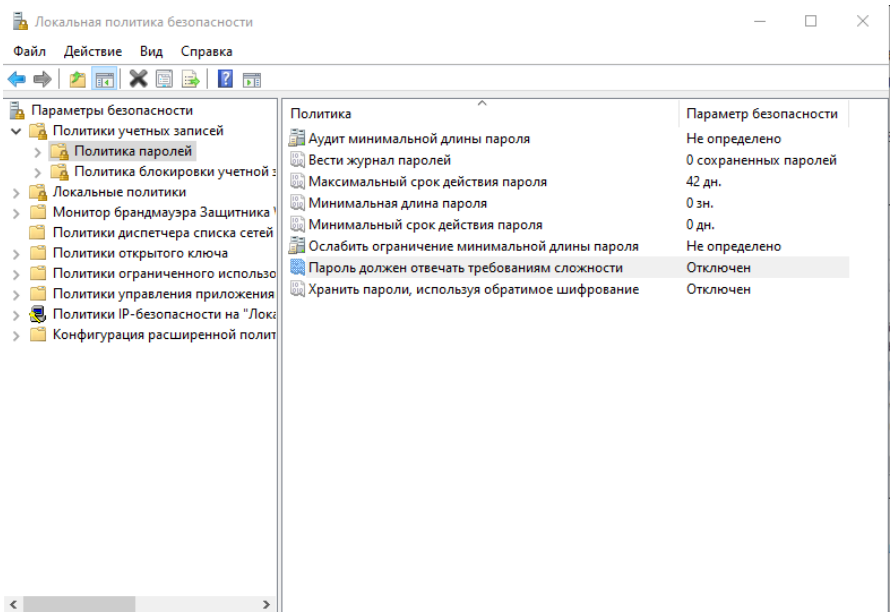

Рис. 4. Окно настройки политики паролей в Windows

Таким образом, анализ существующих подходов к построению верификатора стойкости паролей показывает, что наиболее распространенным решением является выделение правил, в которых акцент делается прежде всего на длину пароля и его наличие в имеющихся словарях паролей. Такое решение обладает рядом недостатков, в т.ч. сложностью реализации более комплексных требований, отражающих свойства высокой стойкости паролей. В отличие от этого, построение верификатора с использованием машинного обучения позволяет алгоритмам самим формулировать эти требования в произвольно комплексной форме, отталкиваясь только от инцидентов – имеющихся для каждой категории стойкости списков известных паролей. Одним из свойств указанного подхода является его универсальность, т.к. под каждый конкретный случай не требуется создавать и настраивать политику паролей (количество специальных символов, количество цифр и т.д.). Машина имитирует смесь многих алгоритмов проверки надежности пароля. Обученную машину можно использовать как отдельный модуль (например, PAM-модуль в Linux), который всегда будет точно определять стойкость паролей. Например, в случае если разработчик создает какую-либо новую социальную сеть, ему будет достаточно обучить машину на основании датасета, собранного из других популярных социальных сетей, где политика паролей считается эффективной: TikTok[4], Twitter[5] (запрещена в Российской Федерации), Facebook[6] (запрещена в Российской Федерации) и т.п. После этого машина встраивается в виде верификатора, который оценивает пароль и выдает соответствующую информацию пользователю при регистрации. В таком случае, если введенному паролю будет поставлен высший балл стойкости, пользователь может быть уверен в том, что его пароль действительно стойкий без привязки к конкретным требованиям и рекомендациям.

МЕТОДОЛОГИЯ И РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТА

Задача верификации стойкости пароля может быть представлена как задача классификации, где объектом данных является пароль, а классом – его уровень стойкости.

В качестве набора данных для эксперимента использовался датасет одной из самых крупных утечек паролей – хостинга 000webhost [8]. С помощью инструмента *PARS* [9], в котором находятся множество счетчиков, оценивающих стойкость пароля, независимым разработчиком была создана база, содержащая пароли и их оценку на основании трех разных алгоритмов, реализованных компаниями Twitter, Microsoft[7], Battle.net[8]. Сам датасет содержит в себе лишь те пароли, которые были оценены всеми тремя

[4] https://www.tiktok.com/. Дата обращения 01.02.2023. / Accessed February 01, 2023.

[5] https://twitter.com/. Дата обращения 01.02.2023. / Accessed February 01, 2023.

[6] http://facebook.com/. Дата обращения 01.02.2023. / Accessed February 01, 2023.

[7] https://www.microsoft.com/. Дата обращения 01.02.2023. / Accessed February 01, 2023.

[8] https://www.battle.net. Дата обращения 01.02.2023. / Accessed February 01, 2023.

измерителями одинаково. В наборе представлено три класса оценки пароля (0 – низкий, 1 – средний, 2 – высокий). Датасет содержит в себе несбалансированные данные: 496649 паролей с оценкой «1», 89662 паролей с оценкой «0», 83113 сильных паролей.

Примеры паролей для каждого уровня стойкости представлены в табл. 1. Распределения длины пароля в целом по набору данных и по отдельным уровням стойкости представлены на рис. 5 и 6.

Таблица 1. Примеры паролей из учебного набора данных

Стойкость пароля	Примеры паролей
0	wewes19 asdas95
1	oyeleye1 80188063JA
2	JFRTgxTQyNQTh9ZD d7a6AoTMxMw0dLVy

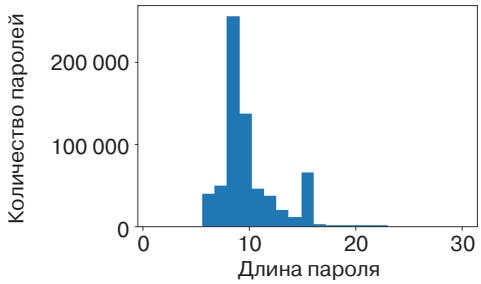

Рис. 5. Гистограмма длины пароля в наборе данных

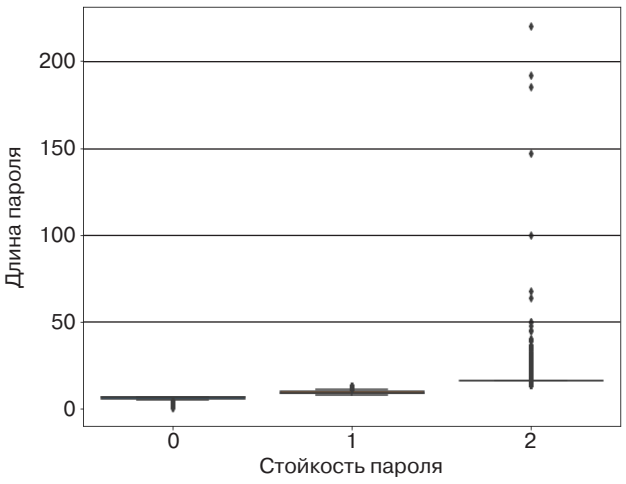

Рис. 6. Распределение длины пароля в зависимости от стойкости в наборе данных

Для проведения эксперимента набор после проведения случайного перемешивания разбивался на обучающий (80%) и тестовый (20%).

Для представления текстового пароля в виде вектора числовых значений в целях учета не только наличия, но и веса каждого отдельного символа, использовался метод TF-IDF (term frequency-inverse document frequency) [10]:

$$TF = \frac{\text{число вхождений символа в пароль}}{\text{общее количество символов в пароле}},$$

$$IDF = \log_2 \frac{\text{общее количество паролей}}{\text{количество паролей, в которых встречается символ}},$$

$$TFIDF = TF \cdot IDF,$$

После применения к набору паролей метода TF-IDF был получен набор, где каждому паролю соответствует вектор длины, совпадающей с размером словаря, а каждому элементу вектора — вес TF-IDF-символа, порядковый номер которого в словаре соответствует порядковому номеру элемента в векторе.

Для обучения классических алгоритмов использовалось уравнивание количества экземпляров в каждом классе с использованием технологии undersampling, для выбора гиперпараметров — k-fold кросс-валидация [11] с $k = 5$. Полученные результаты обучения алгоритмов на обучающем наборе данных, оцененные с использованием макро f1-меры [12], представлены в табл. 2.

Таблица 2. Результаты обучения, оцененные на обучающем наборе методом кросс-валидации

Используемый алгоритм	Найденное значение гиперпараметра	Значение макро f1-меры
Метод опорных векторов	Коэффициент регуляризации: 1	0.806
Случайный лес	Максимальная глубина дерева: 32	0.903
Бустинг	Максимальная глубина дерева: 32	0.946

Вместе с тем, использование классических методов машинного обучения осложняется необходимостью преобразования паролей с использованием алгоритма TF-IDF, что является затратной процедурой и требует пересчета весов при обновлении набора паролей. Кроме этого, указанные алгоритмы представляют пароли в виде неупорядоченного набора символов (упрощение, используемое в подходе «Bag of Words»[9]) и не учитывают взаимное расположение

[9] Упрощенное представление текста в виде мешка (мультимножества) его слов без какого-либо учета грамматики и порядка слов, но с сохранением информации об их количестве. [A simplified representation of the text as a bag (multiset) of its words without any regard to grammar or word order but with retention of information about their number.]

символов. Это частично может быть исправлено использованием моделей биграмм и триграмм, что в свою очередь значительно усложнит процесс обучения. В отличие от этого, рекуррентные нейронные сети работают напрямую с последовательностями символов произвольной длины [13]. Так, для рекуррентной нейронной сети пароли «PASSWORD» и «AWSSODPR» будут определяться двумя разными векторами, тогда как при использовании TF-IDF – одним. Поэтому для оценки стойкости пароля в эксперименте в качестве метода, альтернативного классическим алгоритмам машинного обучения, использовалась также реализованная с использованием библиотеки PyTorch [14] нейронная сеть, состоящая из следующих слоев:

1. Слой embedding, используемый для преобразования пароля, состоящего из символов, в вектор числовых значений.
2. Слой long short-term memory (LSTM) – особая разновидность архитектуры рекуррентных нейронных сетей, способных к обучению долговременным зависимостям, что важно при работе с паролями большой длины [15].
3. Линейный слой, используемый для преобразования внутреннего состояния LSTM-слоя в оценки принадлежности пароля к категориям стойкости (category scores).

Результаты сравнения работы алгоритма бустинга и рекуррентной LSTM-сети на тестовом наборе приведены в табл. 3.

Таблица 3. Результаты обучения, оцененные на тестовом наборе

Показатель	Бустинг	Рекуррентная LSTM-сеть
Верность	0.972	0.9995
Точность	0.971	0.9994
Полнота	0.971	0.9990
Макро f1-мера	0.971	0.9992

Исходный код эксперимента можно увидеть на сайте[10].

Следует отметить, что используемый в эксперименте подход никак не снижает возможности хакера в случае применения фишинговых атак, атак кейлоггеров, а также атак «злоумышленник посередине». Однако, если обратиться к исследованию Data Breach Investigation Report от 2020 г. [16], то можно отметить, что 89% всех разновидностей взлома связаны с каким-либо видом злоупотребления учетными данными (атаки грубой силы и их подвиды, а также атаки, нацеленные на повторное использование людьми учетных данных). Данный факт говорит о том, что подход к определению стойкости пароля с помощью машинного обучения минимизирует риски по большинству векторов возможных атак, что также объясняет значимое количество современных исследований по этому направлению [17–21].

Таким образом, можно сделать вывод о том, что рекуррентная LSTM-сеть опережает традиционные методы машинного обучения, приближая значения показателя качества классификации стойкости паролей к единице.

ЗАКЛЮЧЕНИЕ

В статье предложен подход к построению верификатора стойкости пароля с применением методов машинного обучения, проведено сравнение нескольких алгоритмов на наборе данных паролей, помеченных по уровню стойкости. Показано, что такой подход обладает рядом преимуществ по сравнению с классическими методами верификации стойкости пароля, работающими без применения технологий машинного обучения. Так, использование методов машинного обучения для построения верификатора позволяет формулировать требования к сложности пароля в произвольно комплексной форме, отталкиваясь только от инцидентов. Кроме этого, предложенный в статье подход позволяет лучше противостоять атакам, являющимся подвидом атак грубой силы, а также атакам с помощью радужных таблиц. В первом случае это достигается тем, что сложный пароль, стойкость которого оценена с помощью алгоритма машинного обучения, делает перебор пароля невозможным в связи с огромными временны́ми затратами, которые потребуются злоумышленнику для выполнения этой задачи. Во втором случае атаковать сложный пароль будет возможно, только применив огромную радужную таблицу, которая, в свою очередь, потребует использования значительного объема ресурсов злоумышленника, что делает такую атаку нерелевантной.

Среди рассмотренных методов машинного обучения особенную эффективность продемонстрировали рекуррентные нейронные сети. Стоит отметить, что обучение представления текстового пароля в виде вектора числовых значений, заключающееся в нахождении весов слоя embedding, происходит одновременно с обучением всей сети для максимизации точности классификации. Это позволяет выбирать самой нейронной сети такую векторизацию пароля, которая эффективна именно для решаемой задачи. Кроме этого, нейронные сети работают с последовательностью символов, а не только с их наличием в пароле, что позволяет отказаться от упрощения, используемого в подходе «Bag of Words».

[10] https://github.com/james116blue/password_strength_verifier. Дата обращения 01.02.2023. / Accessed February 01, 2023.

Вышесказанное позволяет выделить использование рекуррентной нейронной сети как одно из наиболее перспективных направлений исследований для построения верификатора стойкости пароля.

Вклад авторов. Все авторы в равной степени внесли свой вклад в исследовательскую работу.

Authors' contribution. All authors equally contributed to the research work.

СПИСОК ЛИТЕРАТУРЫ / REFERENCES

1. Conklin A., Dietrich G., Walz D. Password-based authentication: a system perspective. In: *Proceedings of the 37th Annual Hawaii International Conference on System Sciences*. 2004; IEEE. https://doi.org/10.1109/HICSS.2004.1265412
2. Dell'Amico M., Michiardi P., Roudier Y. Password strength: An empirical analysis. In: *2010 Proceedings IEEE INFOCOM*. 2010; IEEE. https://doi.org/10.1109/INFCOM.2010.5461951
3. Chakrabarti S., Singhal M. Password-based authentication: Preventing dictionary attacks. *Computer*. 2007;40(6):68–74. https://doi.org/10.1109/MC.2007.216
4. Shay R., Komanduri S., Kelley P.G., Leon P.G., Mazurek M.L., Bauer L., Christin N., Cranor L.F. Encountering stronger password requirements: user attitudes and behaviors. In: *Proceedings of the Sixth Symposium on Usable Privacy and Security.* 2010; Article 2. https://doi.org/10.1145/1837110.1837113
5. Селифанов В.В. Оценка эффективности системы защиты информации государственных информационных систем от несанкционированного доступа. *Интеграция науки, общества, производства и промышленности: сборник статей Международной научно-практической конференции*. 2016. С. 109–113. [Selifanov V.V. Evaluation of the efficiency of the information protection system of state information systems from unauthorized access. In: *Integration of Science, Society, production and Industry: Collection of Articles of the International Scientific and Practical Conference*. 2016. P. 109–113 (in Russ.).]
6. Ferreira J.F., Johnson S.A., Mendes A., Brooke P.J. Certified password quality: a case study using Coq and Linux pluggable authentication modules. In: *Integrated Formal Methods. IFM 2017. Lecture Notes in Computer Science.* V. 10510. Springer International Publishing; 2017. P. 407–421. https://doi.org/10.1007/978-3-319-66845-1_27
7. Alshare K.A., Lane P.L., Lane M.R. Information security policy compliance: a higher education case study. *Information & Computer Security*. 2018;26(1):91–108. https://doi.org/10.1108/ICS-09-2016-0073
8. AlSabah M., Oligeri G., Riley R. Your culture is in your password: An analysis of a demographically-diverse password dataset. *Computers & Security*. 2018;77:427–441. https://doi.org/10.1016/j.cose.2018.03.014
9. Ji S., Yang S., Wang T., Liu C., Lee W.H., Beyah R. Pars: A uniform and open-source password analysis and research system. In: *ACSAC' 15*: *Proceedings of the 31st Annual Computer Security Applications Conference.* 2015. P. 321–330. https://doi.org/10.1145/2818000.2818018
10. Aizawa A. An information-theoretic perspective of TF–IDF measures. *Information Processing & Management*. 2003;39(1):45–65. https://doi.org/10.1016/S0306-4573(02)00021-3
11. Bishop C.M. *Pattern Recognition and Machine Learning*. New York: Springer; 2006. 738 p.
12. Lever J., Krzywinski M., Altman N. Classification evaluation: It is important to understand both what a classification metric expresses and what it hides. *Nat. Methods*. 2016;13(8):603–604. https://doi.org/10.1038/nmeth.3945
13. Medsker L.R., Jain L.C. (Eds.). *Recurrent Neural Networks*. *Design and Applications*. CRC Press; 2001. P. 64–67.
14. Imambi S., Prakash K.B., Kanagachidambaresan G.R. PyTorch. In: Prakash K.B., Kanagachidambaresan G.R. (Eds.). *Programming with TensorFlow*. EAI/Springer Innovations in Communication and Computing (book series). Springer; 2021. P. 87–104. https://doi.org/10.1007/978-3-030-57077-4_10
15. Yu Y., Si X., Hu C., Zhang J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput*. 2019;31(7):1235–1270. https://doi.org/10.1162/neco_a_01199
16. Jartelius M. The 2020 Data Breach Investigations Report–a CSO's perspective. *Network Security*. 2020;2020(7):9–12. https://doi.org/10.1016/S1353-4858(20)30079-9
17. Sarkar S., Nandan M. Password Strength Analysis and its Classification by Applying Machine Learning Based Techniques. In: *2022 Second International Conference on Computer Science, Engineering and Applications* (*ICCSEA*). IEEE, 2022. P. 1–5. https://doi.org/10.1109/ICCSEA54677.2022.9936117
18. Sakya S.S., Mauparna M.N. Building a Multi-class Password Strength Generator and Classifier Model by Augmenting Supervised Machine Learning Techniques. Preprint. 2022. https://doi.org/10.21203/rs.3.rs-1820885/v1
19. Murmu S., Kasyap H., Tripathy S. PassMon: A Technique for Password Generation and Strength Estimation. *J. Network Syst. Manage*. 2022;30(1):13. https://doi.org/10.1007/s10922-021-09620-w
20. Tran L., Nguyen T., Seo C., Kim H., Choi D. *A Survey on Password Guessing*. *arXiv preprint arXiv*:2212.08796. 2022. https://doi.org/10.48550/arXiv.2212.08796
21. Xiao Y., Zeng J. Dynamically generate password policy via Zipf distribution. *IEEE Transactions on Information Forensics and Security*. 2022;17:835–848. https://doi.org/10.1109/TIFS.2022.3152357

Об авторах

Беликов Владимир Вячеславович, к.воен.н., доцент, доцент кафедры информационной безопасности № 252 Института искусственного интеллекта ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: belikov_v@mirea.ru. Scopus Author ID 57983605100, https://orcid.org/0000-0003-1423-1072

Прокуронов Иван Андреевич, специалист инженерно-криптографического анализа, ООО «СФБ Лаборатория» (127083, Россия, Москва, ул. Мишина, д. 56, стр. 2). E-mail: miltumultik@gmail.com. https://orcid.org/0000-0003-0999-311X

About the authors

Vladimir V. Belikov, Cand. Sci. (Military), Assistant Professor, Department of Information Security, Institute of Artificial Intelligence, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: belikov_v@mirea.ru. Scopus Author ID 57983605100, https://orcid.org/0000-0003-1423-1072

Ivan A. Prokuronov, Cryptographic Analysis Specialist, SFB Laboratory (56/2, Mishina ul., Moscow, 127083 Russia). E-mail: miltumultik@gmail.com. https://orcid.org/0000-0003-0999-311X