Состязательные атаки на модели обнаружения фишинга на основе URL-адресов

Е.Д. Шустова, О.Р. Лапонина,

Аннотация **– Фишинговые атаки остаются одной из наиболее распространённых и экономически значимых киберугроз. Модели машинного и глубокого обучения для обнаружения фишинга по URL-адресам достигают F1-score 0,95–0,99 на стандартных наборах данных, однако их устойчивость к целенаправленным состязательным (adversarial) модификациям входных данных остаётся недостаточно изученной.**

В настоящей работе предложены систематизация видов фишинговых атак и методов их обнаружения, сравнительный анализ URL-based архитектур: Random Forest, символьная CNN, LSTM, CNN+LSTM, классификация состязательных атак на детекторы фишинга, разработан Python-фреймворк, реализующий семь типов атак уклонения в режиме чёрного ящика: подстановку омоглифов, тайпсквоттинг, инъекцию поддомена, удлинение URL, кодирование URL, GAN-мутацию с лучевым поиском и композитные цепочки.

Разработанный фреймворк удовлетворяет следующим требования: генерация состязательных URL для семи типов атак; CLI с группами параметров для каждого типа; Python API для встраивания в автоматизированные пайплайны; детерминированная генерация через seed-параметр; поддержка предопределённых и пользовательских цепочек; визуальная подсветка изменений в verbose-режиме.

Экспериментальная оценка выполнена на датасете из 73575 URL (PhishTank + Marchal2014). Показано, что омоглифные модификации снижают F1-score на 10–15% у всех архитектур; тайпсквоттинг – на 9–14%; инъекция поддомена – на 8 % у Random Forest и 8–13 % у DL-моделей; удлинение URL – на 10% у Random Forest и до 4% у DL-моделей; кодирование URL – на 6% у Random Forest и 0–10 % у DL-моделей (CNN+LSTM не деградирует). Цепочка full_evasion снижает F1 до 0,79–0,84 у всех архитектур; цепочка maximum – до 0,77–0,84. GAN-мутация с эвристическим scoring снижает F1 на 14–16% без доступа к параметрам модели. Все атаки переносимы в режиме чёрного ящика.

Сформулированы практические рекомендации по повышению устойчивости детекторов: NFKC Unicode-нормализация, IDN/Punycode-детектор, состязательное обучение, ансамблирование Random Forest и DL, ограничение длины URL. Производительность фреймворка составляет до 220000 URL/мин для одиночных атак и 48 000 URL/мин для трёхстадийных цепочек.

Ключевые слова **– Обнаружение фишинга, URL-адрес, состязательные атаки, устойчивость моделей Random Forest, символьная CNN, LSTM, CNN+LSTM, Python-фреймворк, омоглифные модификации, тайпсквоттинг, инъекция поддомена, удлинение URL, кодирование URL, GAN-мутация.**

Статья получена 15 мая 2026.
Е.Д. Шустова – МГУ имени М.В. Ломоносова (email: rit1122014@yandex.ru).
О.Р. Лапонина – МГУ имени М.В. Ломоносова (email: laponina@oit.cmc.msu.ru).

I. Введение

A. Актуальность темы

В современном цифровом мире фишинг остаётся одной из наиболее массовых и экономически значимых форм киберпреступности. По данным «Лаборатории Касперского» за 2024 год, число попыток перехода по фишинговым ссылкам превысило 893 миллиона – прирост в 26% по сравнению с 2023 годом [1]. AntiPhishing Working Group (APWG) зафиксировала в первом квартале 2024 года свыше 1,1 миллиона уникальных фишинговых доменов [2]. Совокупный ущерб от фишинга в 2023–2024 годах оценивается в десятки миллиардов долларов: ФБР сообщает о 18,7 млрд долл. потерь в США за 2023 год [3], а средняя стоимость одного инцидента утечки данных достигла 4,76 млн долл. [4]. Доля фишинга среди всех инцидентов информационной безопасности стабильно удерживается на уровне 30–36 % согласно Verizon DBIR [5].

Качественная эволюция угрозы определяется переходом злоумышленников от массовых спам-рассылок к целенаправленным кампаниям spear-phishing с применением больших языковых моделей (LLM) для генерации персонализированных писем, homograph-атак на доменные имена, fast-flux DNS [6] и клонирования легитимных сайтов в реальном времени.

Для противодействия этим угрозам активно применяются модели машинного обучения и глубокого обучения. URL-based классификаторы на основе Random Forest (RF), сверточных нейронных сетей (CNN) и рекуррентных сетей с долгосрочной памятью (LSTM) достигают F1-score 0,95–0,99 на стандартных наборах данных [7]–[9]. Тем не менее высокая точность на тестовых выборках не гарантирует устойчивости к целенаправленным манипуляциям с входными данными – состязательным (adversarial) атакам [10], [11].

B. Обоснование выбора URL-based подхода

В данной работе рассматриваются атаки, действующие исключительно на уровне URL-адреса. Данный выбор обусловлен рядом архитектурных преимуществ.

Предварительная фильтрация. URL-анализ выполняется до загрузки страницы, что позволяет встраивать детекторы в почтовые шлюзы, DNS-резолверы и сетевые прокси [12].

Масштабируемость. Анализ строки URL занимает микросекунды без сетевых запросов и рендеринга DOM, обеспечивая производительность 105 – 106 записей в секунду [13].

Распространённость. URL-based модели составляют

наибольший класс развёртываемых детекторов фишинга в браузерных расширениях, антивирусных продуктах и DNS-фильтрах [13], [14].

Исследовательская ниша. Несмотря на теоретические описания URL-based атак в литературе [10], [15], систематизированных воспроизводимых инструментов с параметрическим управлением и поддержкой композитных цепочек в открытом доступе не представлено.

C. Вклад статьи

Вкладами данной статьи являются:

1) Систематизированная классификация URL-based состязательных атак на детекторы фишинга с формализацией целевых признаков и профилей уязвимостей для каждого типа атак и каждой архитектуры.
2) Python-фреймворк с открытым исходным кодом, реализующий семь типов состязательных атак (подстановка омоглифов, тайпсквоттинг, инъекция поддомена, удлинение URL, кодирование URL, GAN-мутация с лучевым поиском, композитные цепочки) с параметрическим управлением и seed-based воспроизводимостью – аналоги в открытом доступе не известны.
3) Экспериментальная оценка на датасете 73 575 URL с четырьмя архитектурами (RF, CNN, LSTM, CNN+LSTM), фиксирующая деградацию F1-score и архитектурно-специфические паттерны уязвимостей.
4) Практические рекомендации по повышению устойчивости URL-based детекторов, включая NFKC Unicode-нормализацию, IDN-детектор, состязательное обучение и ассемблирование.

D. Структура статьи

В разделе II рассматриваются виды фишинговых атак и сравниваются методы их обнаружения. В разделе III описываются ML/DL-модели URL-based обнаружения. В разделе IV систематизируются состязательные атаки. В разделе V описывается архитектура и реализация фреймворка. Раздел VI содержит экспериментальное исследование. Раздел VII формулирует рекомендации. Раздел VIII – заключение.

II. ФИШИНГОВЫЕ АТАКИ И МЕТОДЫ ИХ ОБНАРУЖЕНИЯ

A. Определение и классификация фишинга

Фишинг – разновидность социальной инженерии, при которой злоумышленник создаёт ложное доверие к коммуникации или веб-ресурсу с целью принудить жертву к раскрытию конфиденциальных данных либо к установке вредоносного программного обеспечения [14].

По каналу доставки и степени таргетирования выделяют восемь основных разновидностей атак [5], [16].

Email-фишинг реализуется путём массовой рассылки поддельных писем, имитирующих уведомления от банков, государственных органов и популярных сервисов. Письма эксплуатируют срочность («ваш счёт будет заблокирован через 24 часа») и содержат вредоносную гиперссылку.

Spear-phishing представляет собой целенаправленную атаку на конкретного человека или организацию с применением персонализированного контента, собранного из открытых источников. По данным Verizon DBIR 2024, spear-phishing является вектором проникновения в 36% корпоративных инцидентов [5].

Whaling нацелен на топ-менеджеров (CEO, CFO, CISO) с целью инициировать несанкционированные финансовые переводы (Business Email Compromise).

Smishing и вишинг используют SMS и голосовые звонки с подменой номера (caller ID spoofing).

Фарминг перенаправляет пользователей с легитимного домена на поддельный сайт через отравление DNS-кэша или модификацию файла hosts.

Клонированный фишинг: злоумышленник копирует легитимное письмо, заменяя ссылки или вложения на вредоносные аналоги.

Фишинговые сайты имитируют банки, платёжные системы и порталы. Для обмана пользователя на уровне URL применяются: combosquatting – добавление слов к бренду (paypal-secure.com) [17]; typosquatting – эксплуатация опечаток (goooogle.com); IDN homograph-атаки – использование визуально идентичных Unicode-символов [18]; URL-кодирование – замена символов hex-последовательностями.

B. Методы обнаружения фишинговых писем

Правила и фильтры контента. Эвристические правила анализируют адрес отправителя, ключевые слова срочности, заголовки и гиперссылки письма. При актуальных правилах точность составляет 80–90 % [19], [20], однако злоумышленники подбирают тексты, обходящие стандартные паттерны. Протоколы SPF, DKIM и DMARC снижают эффективность спуфинга домена, но не защищают от lookalike-доменов.

Анализ поведения и аномалий. Поведенческий анализ строит модель нормальной активности отправителя и обнаруживает отклонения с помощью методов аномалий [21]. Ограничения: необходимость исторических данных за несколько месяцев, регуляторные требования к метаданным (GDPR).

NLP и LLM-анализ текста. Предобученные модели BERT [22] и DistilBERT, дообученные на фишинговых датасетах, достигают F1 0,96–0,98. С появлением коммерческих LLM злоумышленники получили возможность генерировать грамматически безупречный фишинговый текст, затрудняющий частотный анализ [23].

C. Методы обнаружения фишинговых сайтов

Чёрные и белые списки. Чёрные списки (PhishTank, OpenPhish, Google Safe Browsing) обеспечивают высокую точность для известных угроз при минимальной вычислительной стоимости [24]. Медиана времени от публикации фишинговой страницы до попадания в чёрный список составляет 12–24 часа, тогда как большинство жертв приходит на фишинговые сайты в первые часы. Белые списки применимы в ограниченных корпоративных средах (SCADA, кассы розничной сети), но непрактичны для обычных пользователей.

URL-based и доменный анализ. URL-анализ строится на извлечении признаков из строки URL без загрузки страницы: лексические (длина URL, количество поддоменов, доля специальных символов, ключевые слова); хостовые (возраст домена, WHOIS-данные); сетевые (DNS TTL, записи A, репутация ASN) [7], [25]. Браузерное расширение принимает решение за 300 – 500 мс без загрузки контента страницы [12].

Анализ содержимого страниц и визуальный анализ. HTML-based детекторы [26] извлекают признаки из DOM-структуры, встроенных скриптов, форм и метатегов. Визуальные детекторы (VisualPhishNet [27], Phishpedia [28]) сравнивают скриншот с эталонными шаблонами брендов через CNN-эмбеддинги. Гибридные системы объединяют несколько источников признаков, достигая точности 97–99% [19].

D. Сравнительный анализ методов

Сравнение основных способов обнаружения представлено в таблице 1.

Таблица 1. Сравнение подходов к обнаружению фишинговых сайтов

Подход	Скорость	Точность	Zero-day	Ресурсы	RT
Чёрные списки	Высок.	Высок.*	Низк.	Низк.	Да
URL-based ML	Высок.	95–99%	Средн.	Низк.	Да
HTML-based DL	Средн.	93–97%	Средн.	Средн.	Нет
Визуальный	Низк.	90–96%	Высок.	Высок.	Нет
Гибридный	Низк.	97–99%	Высок.	Высок.	Нет

* Только известные угрозы. RT – применимость в реальном времени.

URL-based методы обеспечивают наилучший баланс скорости, точности и применимости в реальном времени, что делает их основным объектом практической части данной работы.

III. Модели машинного обучения для URL-based обнаружения фишинга

A. Признаки URL-адреса

URL-адрес имеет строго определённую структуру согласно RFC 3986 [29]:

```
scheme://[userinfo@]host[:port]/path[?query][#fragment]
```

Для машинного обучения из этой структуры извлекается набор числовых и булевых признаков трёх групп [7], [25].

Лексические признаки: общая длина URL, длина домена, длина пути, количество поддоменов, наличие символа @, доля цифр и специальных символов, наличие IP-адреса вместо домена, ключевые слова высокого риска (login, secure, verify, bank), тип TLD, hex-кодирование в строке, глубина пути, число query-параметров.

Хостовые признаки: возраст домена (регистрации менее 30 суток подозрительны), анонимный WHOIS, совпадение с брендом, количество перенаправлений.

Сетевые признаки: DNS TTL (низкое значение — признак fast-flux), наличие записей A, количество IP-адресов, нахождение ASN в чёрном списке.

B. Классические алгоритмы машинного обучения

Random Forest. Случайный лес объединяет предсказания T деревьев решений, обученных на бутстрап выборках [30]:

$$\hat{y} = majorityvote\big(h_1(x),\ldots,h_T(x)\big). \quad (1)$$

На датасете 73k URL (PhishTank + Alexa) RF с 48 лексическими признаками достигает F1 = 0,975 [7]. Встроенная оценка важности признаков (Gini impurity) выявляет наиболее информативные индикаторы: длину URL, наличие IP-адреса и количество поддоменов [31].

XGBoost и градиентный бустинг. XGBoost [32] строит ансамбль деревьев последовательно, минимизируя регуляризованную функцию потерь. На комбинированных признаках (URL + гиперссылки страницы) достигает 99% accuracy [13]. Интерпретируемость через SHAP-анализ [33] критически важна для аудита в регулируемых отраслях.

C. Архитектура глубокого обучения

Модели глубокого обучения обрабатывают URL как последовательность символов без ручного извлечения признаков [34]. Словарь включает 97 печатаемых ASCII-символов; строка дополняется или усекается до Lmax = 200 символов.

Символьная CNN. Архитектура: Embedding(97× 64) → три параллельные ветви Conv1D с ядрами 3, 5, 7 и числами фильтров 128, 256, 128 → GlobalMaxPool → Concat → Dense(128, ReLU) → Dropout(0,3) → Dense(1, sigmoid) [35]. Свёрточные ядра различного размера фиксируют локальные паттерны разного масштаба: биграммы, триграммы, пентаграммы символов. PhishGuard на данной архитектуре достигает F1 = 0,982 [8].

LSTM. Рекуррентная архитектура [36]: Embedding → LSTM(256) → LSTM(128) → Dense(64, ReLU) → Dropout(0,3) → Dense(1, sigmoid).

Уравнения вентилей:

$$f_t=\sigma(W_f[h_{t-1},x_t]+b_f),$$
$$i_t=\sigma(W_i[h_{t-1},x_t]+b_i),$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c[h_{t-1}, x_t] + b_c). \quad (2)$$

AntiPhishStack [9] реализует стековую генерализацию LSTM-моделей и достигает F1 = 0,975.

Гибридная CNN+LSTM. Схема: Embedding → Conv1D(128, k=5, ReLU) → MaxPool(2) → LSTM(128) → Dense(64, ReLU) → Dropout(0,3) → Dense(1, sigmoid). Conv1D выступает как детектор n-грамм, снижая размерность; LSTM моделирует дальние зависимости вдоль последовательности [9]. F1 = 0,97–0,99.

GramBeddings и URLNet. GramBeddings [37] объединяет четыре потока n-граммных эмбеддингов (unigram, bigram, trigram, хостовые признаки) через CNN+LSTM с механизмом внимания. Точность 98,27% на 800k URL без ручного инжиниринга признаков.

URLNet [38] применяет двойную CNN на символьном и словном уровне, достигая F1 = 0,976 на PhishTank+Alexa.

D. *Модели, используемые в экспериментальной части*

Для тестирования отобраны четыре архитектуры, представляющие разные парадигмы обнаружения (таблица 2).

Таблица 2. Конфигурации экспериментальных моделей

Модель	Конфигурация	Пар., M
RF	100 деревьев, depth=20, 48 признаков	—
CNN	3 ветви Conv1D (k=3,5,7), emb=64	1,2
LSTM	2 слоя LSTM (256+128), emb=64	2,1
CNN+LSTM	Conv1D(k=5)+LSTM(128), emb=64	2,8

Общие параметры обучения DL-моделей: Adam [39], $\eta = 0{,}001$; batch = 256; epochs = 30; early stopping (patience = 5) по val_loss [40]; разделение 80/20 со стратификацией.

Базовые метрики на тестовой выборке приведены в таблице 3.

Таблица 3. Базовые метрики моделей до атак (тестовая выборка)

Модель	Prec.	Recall	F1	AUC
RF	0,930	0,910	0,920	0,971
CNN	0,970	0,950	0,960	0,989
LSTM	0,960	0,940	0,950	0,983
CNN+LSTM	0,970	0,960	0,970	0,992

IV. Состязательные атаки на детекторы фишинга

A. *Формальное определение и таксономия*

Состязательная атака – целенаправленное изменение входных данных x, при котором классификатор f выдаёт ошибочное предсказание при сохранении семантики объекта [41]. Формально требуется найти $x' = x + \delta$ такой, что:

$$f(x') \neq f(x),\ \|\delta\|_p \leq \varepsilon, \tag{3}$$

где δ – пертурбация, ε – ограничение нормы, $p \in \{0, 1, 2, \infty\}$. Для URL x' должен оставаться функционально корректным адресом и не вызывать подозрений у пользователя.

По фазе атаки различают: атаки уклонения (evasion) – воздействие на развёрнутую модель без изменения её параметров (сценарий настоящей работы); и атаки отравления (poisoning) – внедрение искажённых примеров в обучающую выборку [42].

По осведомлённости атакующего: белый ящик (white-box) – полный доступ к архитектуре и весам, методы FGSM [41], PGD [42] и C&W [43]; серый ящик – известен тип модели, но не параметры; чёрный ящик (black-box) – доступны только входы и выходы, реалистичная модель угрозы для промышленных систем.

Фреймворк настоящей работы реализует атаки уклонения в режиме чёрного ящика: пертурбации формируются без запросов к модели, обеспечивая переносимость на любые детекторы.

B. *Атаки на визуальные и HTML-based модели*

Визуальные детекторы (VisualPhishNet [27], Phishpedia [28]) уязвимы к пиксельным пертурбациям FGSM/PGD на скриншотах [44] и к модификациям SVG-логотипов. Цветовой сдвиг $\Delta H \leq 5^\circ$ в пространстве HSV снижает точность VisualPhishNet с 91 до 67% [45].

HTML-based модели уязвимы к символьной обфускации (Hangul Filler U+3164, Zero-Width Space U+200B) [46], [47], инъекции доброкачественного контента [48], LLM-перефразировке [49] и HTML-обфускации элементы).

C. *Состязательные атаки на URL-based модели*

Подстановка омоглифов. Омоглифная атака эксплуатирует визуальную идентичность символов из разных кодовых таблиц Unicode [18]. Кириллическая «а» (U+0430) неотличима от латинской «а» (U+0061) в большинстве шрифтов, однако является другим символом с точки зрения строкового сравнения и символьного эмбеддинга.

Пусть H(c) – множество гомоглифов символа c, $\alpha \in [0,1]$ – коэффициент интенсивности. Атака:

$$x_i' = \begin{cases} h \sim \mathcal{H}(x_i) & \text{с вероятностью } \alpha, \\ x_i & \text{иначе.} \end{cases} \tag{4}$$

Целевые признаки. Символьный embedding DL-моделей (символы из другой кодовой страницы проецируются на иные позиции); лексические признаки RF (ключевые слова «secure», «login» не обнаруживаются, если слово смешано с гомоглифами).

Тайпсквоттинг. Тайпсквоттинг имитирует опечатки при наборе доменного имени [50]. Четыре стратегии: дублирование символа (goooogle.com), пропуск (gogle.com), транспозиция (googel.com), замена соседней клавишей QWERTY (googke.com). Целевые признаки: точные текстовые совпадения с брендами, n-граммные паттерны доменных имён.

Инъекция поддомена. Атака добавляет название легитимного бренда как поддомен к домену атакующего: google.secure-login.com [10]. Три стиля: bare (paypal.attacker.com), hyphenated (paypal-com.attacker.com), добавление префикса (login.paypal.attacker.com). Целевые признаки: структура строки домена (DL-модели), счётчик поддоменов (RF).

Удлинение URL. В адрес добавляются реалистичные query-параметры (utm_source, session_id, tracking_id), увеличивающие длину строки и число параметров. Генераторы значений: hex, UUID4, base64, Unix-timestamp. Целевые признаки: длина URL и число параметров (RF). DL-модели с усечением до Lmax = 200 практически невосприимчивы [10].

Кодирование URL. Атака заменяет символы hex-последовательностями %XX [29]. Пример:

```
/login?user=admin →
/login?%75%73%65%72=%61%64%6D%69%6E
```

Поддерживается три режима регистра hex-цифр: upper, lower, mixed (случайный выбор). Целевые признаки: доля %-символов (RF), ключевые слова-индикаторы (RF), символьное распределение (DL).

Композитные цепочки. Последовательное применение нескольких атак: каждая разрушает собственную группу признаков [10]. Предопределённые

цепочки: stealth (только омоглифы); typo_homoglyph (тайпо → омоглифы); full_evasion (тайпо → омоглифы → удлинение); encoded_evasion (кодирование → удлинение); maximum (тайпо → омоглифы → кодирование → удлинение → инъекция ПД).

GAN-мутация с лучевым поиском. Метод [51], [52] реализует оптимизационный поиск в пространстве символов без обучения нейросети. Лучевой поиск за G поколений расширяет пул кандидатов тремя операторами мутации: (1) гомоглифный (замена символов высокосходными Unicode-аналогами), (2) символьный (дублирование, удаление, перестановка в netloc), (3) частичное кодирование ASCII-символов пути через %XX. Оценка кандидатов производится через victim_fn (black-box запрос к классификатору) или встроенную лексическую эвристику. Алгоритм:

$$beam_{g+1} = p_K\{(s(m), m) \vee m \in mutate(beam_g)\}, \quad (5)$$

где K – ширина луча, s(·) – целевая функция. Скорость: ≈ 3500 URL/мин против нескольких секунд/URL у нейросетевых GAN [51], [53].

D. *Сравнение типов атак*

Сравнение типов состязательных атак приведено в таблице 4.

Таблица 4. Сравнение URL-based состязательных атак

Атака	Целевые признаки	Уязв. арх.	Зам.
Омоглифы	Embedding, лексика	Все	Низк.
Тайпсквот.	Домен, n-граммы	Все	Низк.
Инъекция ПД	Структура домена	DL	Средн.
Удлинение	Длина URL, параметры	RF	Низк.
Кодирование	Лексика, ключевые слова	RF, DL	Низк.
Композитная	Все группы	Все	Средн.
GAN-мутация	Embedding, лексика	Все	Низк.

V. Архитектура и реализация фреймворка

A. *Требования к системе*

Функциональные требования. (1) генерация состязательных URL для семи типов атак; (2) CLI с группами параметров для каждого типа; (3) Python API для встраивания в автоматизированные пайплайны; (4) детерминированная генерация через seed-параметр; (5) поддержка предопределённых и пользовательских цепочек; (6) визуальная подсветка изменений в verbose-режиме.

Нефункциональные требования. (1) производительность не менее 10 000 URL/мин для одиночных атак; (2) модульность: каждый тип атаки – независимый компонент; (3) расширяемость через принцип Open/Closed; (4) автономность: без внешних API и сетевых запросов; (5) совместимость: Python 3.10+, только tldextract как внешняя зависимость.

B. *Компонентная архитектура*

Фреймворк состоит из четырёх уровней: (1) CLI/API – точка входа python -m src.attacks, вспомогательные функции attack_url() и typosquat_url(); (2) модуль атак src/attacks/ – семь классов атак и базовый абстрактный класс; (3) утилиты src/utils/ – подсветка diff, детектор поддержки цвета терминала; (4) данные homoglyphs.py – словарь из ≈180 пар гомоглифов без логики.

Применяются два паттерна проектирования [54].

Стратегия (Strategy). Каждый класс атаки реализует интерфейс BaseAttack с методами generate() → List[str] и get_attack_info() → Dict; добавление нового типа атаки сводится к созданию нового подкласса.

Цепочка обязанностей. CompositeAttack применяет список BaseAttack последовательно; выход одного звена – вход следующего.

C. *Реализация классов атак*

BaseAttack (абстрактный):

Листинг 1. Базовый класс атаки

```
class BaseAttack(ABC):
    def __init__(self, url: str,
                 count: int = 10,
                 seed: int | None = None):
        self.url = url
        self.count = count
        self.rng = random.Random(seed)

    @abstractmethod
    def generate(self) -> list[str]: ...

    @abstractmethod
    def get_attack_info(self) -> dict: ...
```

SubstitutionAttack хранит словарь гомоглифов трёх уровней сходства (high/medium/low). Метод `generate()` выбирает позиции замены через `random.sample()` без повторений. Параметр `domain_only` ограничивает замены хост-частью URL.

TyposquattingAttack реализует четыре опечатки через перечисление TypoType. QWERTY-матрица смежности определяется статически для раскладок QWERTY и QWERTZ.

SubdomainInjectionAttack использует базу из 50 наиболее часто имитируемых брендов. Реконструкция URL через `urllib.parse.urlunparse()` сохраняет path, query и fragment.

LengthPaddingAttack генерирует значения параметров через четыре формата: hex, uuid4, base64, timestamp.

URLEncodingAttack кодирует выбранные компоненты URL (domain, path, query, fragment). Режим mixed порождает до count уникальных вариантов через псевдослучайный выбор регистра hex-цифр.

GANMutationAttack запускает лучевой поиск за G поколений с шириной луча K. Три оператора мутации: (1) _mutate_homoglyph – замена символов гомоглифами уровня HIGH; (2) _mutate_char_level – дублирование, удаление, перестановка в netloc; (3) _mutate_encoding_partial – %XX-кодирование ASCII-букв пути со случайным регистром. Сложность: O(G · K · |x|).

CompositeAttack реализует паттерн «цепочка обязанностей»:

Листинг 2. Алгоритм CompositeAttack.generate()

```
    pool = [url]
    for attack in chain:
        next_pool = []
        for u in pool:
            attack.url = u
            next_pool.extend(attack.generate())
        pool = rng.sample(next_pool,
                          min(count, len(next_pool)))
    return pool[:count]
```

D. CLI и Python API

Точка входа:

`python -m src.attacks <attack_type> [параметры].`

Пример омоглифной атаки:

Листинг 3. CLI: омоглифная атака

```
$ python -m src.attacks http://paypal-secure.com/
    login \
    substitution --coefficient 0.3 \
    --quality high --count 3 --seed 42 --verbose
Original: http://paypal-secure.com/login
1. http://paypal-secure.com/login   [a->]
2. http://paypal-secure.com/login   [a->]
3. http://paypal-secure.com/login   [o->]
```

Python API для интеграции:

Листинг 4. Python API: композитная атака

```
    composite = CompositeAttack.from_preset(
        url="http://paypal-secure.com/login",
        preset_name="full_evasion",
        count=5, seed=42
    )
    results = composite.generate_with_trace()
```

E. Характеристики производительности

Все атаки реализованы в чистом Python без I/O и сетевых запросов. Производительность (Intel Core i7-12700H, 1 поток, count=10): омоглифы – 185000 URL/мин; тайпсквоттинг – 210000; инъекция ПД – 195000; удлинение – 170000; кодирование – 220 000; GAN-мутация (G = 20, K = 30) – 3 500; composite full_evasion – 48000 URL/мин.

Требование ≥10000 URL/мин выполнено с запасом 4,8–21× для всех типов, кроме GAN-мутации (лучевой поиск). GAN-режим масштабируется через multiprocessing.Pool.

VI. Экспериментальное исследование

A. Датасет и предобработка

Экспериментальный датасет агрегирован из двух источников. PhishTank – 37175 верифицированных фишинговых URL, собранных за 2022–2024 годы. Верификация сообществом с 5-голосным консенсусом обеспечивает низкую долю ложно-позитивных меток. Marchal2014 (PhishStorm) [55] – 36 400 легитимных URL, отфильтрованных по критерию активности домена (ответ 200 OK на момент сбора).

Итоговый датасет: 73 575 URL (37 175 фишинговых – 50,5%; 36400 легитимных – 49,5%). Разделение: 80% обучение / 20% тест со стратификацией; контроль domain leakage – ни один registrable domain не встречается одновременно в обоих разделах.

Предобработка для DL-моделей: перевод в нижний регистр, удаление протокола, усечение до Lmax = 200 символов, кодирование через словарь 97 ASCII-символов. Для RF: 48 лексических и хостовых признаков без сетевых запросов.

B. Параметры атак

Конфигурация атак в эксперименте отображена в таблице 5.

C. Протокол эксперимента

Атаки применялись к 20% фишинговых URL тестовой выборки (0,2 × 7 435 = 1 487 URL). Состязательные варианты замещали оригинальные фишинговые URL в тестовом наборе. Модели переоценивались на модифицированном наборе без переобучения. Первичная метрика – F1-score бинарной классификации.

D. Результаты тестирования

F1-score после атак по типам. Омоглифные модификации показали наибольшее снижение F1 для всех архитектур (RF: −15 %, CNN: −12 %, LSTM: −13 %, CNN+LSTM: −10 %). Этот эффект объясняется тем, что символьный embedding-слой DL-моделей не обучался на кириллических/греческих заменах, а RF не классифицирует кириллицу как «подозрительный» символ в стандартном наборе признаков [10].

Детализация под омоглифной атакой. Снижение Recall значительнее снижения Precision: модели пропускают фишинговые URL с омоглифами (False Negatives растут) при относительно стабильном уровне ложных тревог.

Результаты GAN-мутации. Лучевой поиск с G = 20, K = 30 сходится к вариантам, комбинирующим гомоглифы в netloc и частичное %XX-кодирование пути, нарушая признаки нескольких групп одновременно.

Результаты композитных атак. Цепочка maximum снижает F1 до уровня 0,77–0,84 у всех архитектур, обеспечивая наибольшую деградацию среди рассмотренных вариантов цепочек.

VII. Анализ результатов и рекомендации

A. Анализ паттернов уязвимостей

Омоглифы и тайпсквоттинг – наибольший эффект. Омоглифная замена при α = 0,3 вызывает падение F1 на 10–15% у всех моделей. DL-модели обучались на ASCII-URL; кириллические/греческие замены проецируются на иные или нулевые позиции в пространстве эмбеддингов. RF теряет чёткость в счётчике подозрительных символов: кириллица не считается «подозрительным» символом в стандартном 48-признаковом наборе. Тайпсквоттинг показывает схожую эффективность (−9–14%), нарушая n-граммные паттерны доменных имён [10].

Инъекция поддомена – асимметричный эффект. RF теряет лишь 8 % F1 (счётчик поддоменов как один из 48 признаков), тогда как DL-модели деградируют на 8– 12 %: инъекция «paypal» перед доменом кардинально меняет всю символьную последовательность строки хоста, нарушая паттерны, усвоенные при обучении [11].

Удлинение – избирательный эффект. DL-модели практически невосприимчивы (CNN и CNN+LSTM – 2 %, LSTM – 4 %): усечение до Lmax = 200 выводит добавленные параметры за пределы окна ввода. RF теряет 10% F1: признаки «длина URL» и «число

параметров» являются информативными, и их экстремальные значения нарушают решающие правила деревьев [10].

Кодирование URL – двойной механизм деградации. Для RF: доля %-символов резко возрастает, ключевые слова (login, secure) не совпадают со словарными паттернами, длина URL увеличивается в 2,5–3 раза. Для DL: символы % и hex-цифры вытесняют алфавитный контент, сдвигая распределение входной последовательности от обучающего [10].

CNN+LSTM – лучшая базовая точность и относительно высокая устойчивость. Гибридная архитектура показывает наилучшую базовую точность (F1 = 0,970) и в большинстве сценариев сохраняет устойчивость: под кодированием URL не деградирует (F1 = 0,970), под омоглифами теряет лишь 10%, под инъекцией поддомена – 8 % – меньше, чем CNN и LSTM. Тем не менее под тайпсквоттингом CNN+LSTM уступает CNN и LSTM (0,850 против 0,860) – более высокое базовое качество не гарантирует adversarial-устойчивости во всех сценариях.

Взаимоусиление в композитных цепочках. Цепочка full_evasion снижает F1 до 0,79–0,84. Каждая атака разрушает ту группу признаков, на которую могла бы «опереться» модель после предыдущей атаки – признаки перестают дополнять друг друга при детектировании.

Переносимость (transferability). Все четыре архитектуры демонстрируют схожую деградацию (±2 % от среднего) при каждом типе атаки. Это подтверждает высокую переносимость rule-based атак в режиме чёрного ящика [33].

B. Рекомендации по повышению устойчивости

Нормализация Unicode. Перед анализом URL следует приводить все символы к стандартной форме (NFKC), что устраняет большинство вариантов «невидимых» подмен символов. Для защиты от кириллических омоглифов (визуально неотличимых от латиницы) дополнительно применяется таблица визуально схожих символов Unicode confusables [10].

Проверка доменных имён на нестандартные символы. Доменные имена с символами из нелатинских алфавитов преобразуются в стандартный ASCII-вид (Punycode, RFC 3492 [56]), что позволяет надёжно обнаруживать омоглифные домены вида paypal.com (с кириллической «а»).

Состязательное дообучение. Включение сгенерированных фреймворком примеров атак в обучающую выборку повышает устойчивость модели на 5–10 % [57].

Комбинирование классических и нейросетевых моделей. Объединение предсказаний Random Forest и DL-моделей компенсирует слабые стороны каждого подхода: атаки, эффективные против RF (удлинение URL, кодирование), слабо влияют на DL, и наоборот.

Ограничение длины URL. Простой порог – отклонять URL длиннее 300 символов или с более чем 10 query-параметрами – надёжно блокирует атаки удлинения без ложных срабатываний на легитимных URL.

VIII. Заключение

В настоящей работе проведена систематизация видов фишинговых атак и методов их обнаружения. Установлено, что URL-based методы обеспечивают наилучший баланс скорости (до 106 URL/с), точности (F1 = 0,95 – 0,99) и применимости в реальном времени, однако модели демонстрируют значимую уязвимость к состязательным модификациям входных данных.

Разработан Python-фреймворк, реализующий семь типов состязательных атак в режиме чёрного ящика с единым интерфейсом BaseAttack, CLI, Python API, seed-based воспроизводимостью и шестью предопределёнными цепочками. Производительность составляет до 220000 URL/мин для одиночных атак и 48 000 URL/мин для трёхстадийных цепочек.

Экспериментальная оценка на 73575 URL выявила архитектурно-специфические паттерны уязвимостей: RF наиболее уязвим к удлинению URL (−10% через length-based признаки); CNN и LSTM – к омоглифам и кодированию URL (через символьные эмбеддинги); CNN+LSTM сохраняет устойчивость к кодированию URL (нет деградации) и показывает наименьшие потери под омоглифами и инъекцией поддомена, однако под GAN-мутацией теряет 14 %. Все атаки переносимы: разработанные без доступа к весам моделей, они одинаково эффективны против всех четырёх архитектур.

Цепочка максимально снижает F1 всех моделей до уровня 0,77–0,84, что подтверждает критическую важность мер противодействия: Unicode-нормализации, IDN-детектора, состязательного обучения и ансамблирования.

Благодарности

Авторы благодарны сотрудникам кафедры Информационной безопасности (ИБ) за обсуждения и полезные замечания. Вопросы использования Искусственного интеллекта в кибербезопасности являются одним из основных научных направлений кафедры ИБ факультета ВМК МГУ имени М.В. Ломоносова и рассматривались во множестве магистерских диссертаций и научных работ [58, 59, 60].

Библиография

[1] Kaspersky Lab, “Kaspersky security bulletin 2024: Spam and spam-and-phishing-in-2024, 2024, accessed: 2025-01-10.

[2] Anti-Phishing Working Group, “Phishing activity trends report q1 2024,” https://apwg.org/trendsreports/, 2024, accessed: 2025-01-15.

[3] Federal Bureau of Investigation, “Internet crime report 2023,”https://www.ic3.gov/Media/PDF/AnnualReport/2023_IC3Report.pdf, 2024, accessed: 2025-01-20.

[4] IBM Security, “Cost of a data breach report 2024,” https://www.ibm.com/reports/data-breach, 2024, accessed: 2025-01-20.

[5] Verizon, “2024 data breach investigations report,” https://www.verizon.com/business/resources/reports/dbir/, 2024, accessed: 2025-01-15.

[6] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, “Measuring and detecting fast-flux service networks,” in Network and Distributed System Security Symposium (NDSS), 2008.

[7] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, “Machine learning based phishing detection from URLs,” Expert Systems with Applications, vol. 117, pp. 345–357, 2019.

[8] S. Y. Yerima and M. K. Alzaylaee, “PhishGuard: A convolutional neural network based model for detecting phishing URLs with explainability analysis,” arXiv preprint arXiv:2404.17960, 2024.

[9] A. Anand, S. Garg, G. Choudhary, and N. Pandey, "AntiPhishStack: LSTM-based stacked generalization model for optimized phishing URL detection," Symmetry, vol. 16, no. 2, p. 248, 2024.

[10] G.Apruzzese,P.Laskov,andA.Tastemirova,"SpacePhish:The evasion-space of adversarial attacks against phishing website detectors using machine learning," in Annual Computer Security Applications Conference (ACSAC), 2022, pp. 578–592.

[11] B. Liang, J. Su, W. Guo, Y. Shi, and X. Zhang, "Directed adversarial sampling attacks on phishing detection," Journal of Computer Security, vol. 29, no. 1, pp. 1–29, 2021.

[12] R. Liu, Y. Lin, X. Yang, J. Y. Ng, D. M. Divakaran, and J. S. Dong, "PhishIntel: Toward practical deployment of reference-based phishing detection," in ACM SIGSAC Conference on Computer and Communications Security (CCS), 2024.

[13] G. Apruzzese, M. Colajanni, L. Ferretti, and M. Marchetti, "Revisiting the performance of machine learning-based phishing URL detection," IEEE Transactions on Network and Service Management, vol. 21, no. 2, pp. 1631–1648, 2024.

[14] A.SafiandS.Singh,"Asystematicliteraturereviewonphishing website detection techniques," Journal of King Saud University – Computer and Information Sciences, vol. 35, no. 2, pp. 590– 611, 2023.

[15] G. Apruzzese, P. Laskov, and A. Tastemirova, " Multi-SpacePhish: Extending the evasion-space of adversarial attacks against phishing website detectors using machine learning," ACM Transactions on Privacy and Security, 2023.

[16] N. S. Afanasyeva, D. A. Elizarov, and T. A. Myznikova, "Klassifikatsiya fishingovykh atak i mery protivodeystviya im," Informatsionnaya bezopasnost' regionov, no. 5(89), 2022.

[17] P. Kintis, N. Miramirkhani, C. Lever, Y. Chen, R. Romero-Gomez, N. Pitropakis, N. Provos, and M. Antonakakis, "Hiding in plain sight: A longitudinal study of combosquatting abuse," in ACM SIGSAC Conference on Computer and Communications Security (CCS), 2017, pp. 569–586.

[18] T. Holgers, D. E. Watson, and S. D. Gribble, "Cutting through the confusion: A measurement study of homograph attacks," in USENIX Annual Technical Conference (ATC), 2006, pp. 261–266.

[19] M. A. Adebowale, K. T. Lwin, E. Sanchez, and M. A. Hossain, "Intelligent phishing website detection system using modified frequent pattern-growth," International Journal of Engineering and Advanced Technology, vol. 10, no. 3, pp. 22–29, 2021.

[20] L. Tang and Q. H. Mahmoud, "A survey of machine learning-based solutions for phishing website detection," Machine Learning and Knowledge Extraction, vol. 3, no. 3, pp. 672–694, 2021.

[21] G. Apruzzese, M. Colajanni, L. Ferretti, and M. Marchetti, "Evasion attacks against banking fraud detection systems," https://arxiv.org/abs/2004.06954, 2020, arXiv:2004.06954.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[23] T. Koide, D. Chiba, and M. Akiyama, "Detecting phishing sites using ChatGPT," arXiv preprint arXiv:2306.05816, 2024.

[24] S. Bell and P. Komisarczuk, "An analysis of phishing blacklists: Google Safe Browsing, OpenPhish, and PhishTank," in Australasian Computer Science Week Multiconference (ACSW), 2020.

[25] D.Sahoo,C.Liu,andS.C.Hoi,"MaliciousURLdetectionusing machine learning: A survey," arXiv preprint arXiv:1701.07179, 2017.

[26] Y. Zhang, J. Hong, and L. Cranor, "CANTINA: A content-based approach to detecting phishing web sites," in Proceedings of the 16th International Conference on World Wide Web (WWW), 2007, pp. 639–648.

[27] S. Abdelnabi, K. Krombholz, and M. Fritz, "VisualPhishNet: Zero-day phishing website detection by visual similarity," in Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS), 2020, pp. 1681–1698.

[28] Y. Lin, R. Liu, D. M. Divakaran, J. Y. Ng, Q. Z. Chan, Y. Lu, Y. Li, T. Wang, and J. S. Dong, "Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages," in USENIX Security Symposium, 2021, pp. 3793–3810.

[29] T. Berners-Lee, R. Fielding, and L. Masinter, "RFC 3986: Uniform resource identifier (URI): Generic syntax," IETF. https://www.rfc-editor.org/rfc/rfc3986, 2005.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[31] R. S. Rao and A. R. Pais, "Detection of phishing websites using an efficient feature-based machine learning framework," Neural Computing and Applications, vol. 31, no. 8, pp. 3851–3873, 2019.

[32] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794.

[33] Y. Wang, J. Zhang, and H. Li, "Comprehensive evaluation of adversarial perturbations against ML-based ethereum phishing detection systems," ACM Transactions on the Web, 2025.

[34] P. Yang, G. Zhao, and P. Zeng, "Phishing website detection based on multidimensional features driven by deep learning," IEEE Access, vol. 7, pp. 15 196–15 209, 2019.

[35] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in Advances in Neural Information Processing Systems (NeurIPS), vol. 28, 2015, pp. 649–657.

[36] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[37] A. S. Bozkir, F. C. Dalgic, and M. Aydos, "GramBeddings: A new neural network for URL-based malicious web site detection through n-gram embeddings," Computers & Security, vol. 124, p. 103016, 2023.

[38] H. Le, Q. Pham, D. Sahoo, and S. C. Hoi, "URLNet: Learning a URL representation with deep learning for malicious URL detection," https://arxiv.org/abs/1802.03162, 2018, arXiv:1802.03162.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in International Conference on Learning Representations (ICLR), 2015.

[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, vol. 15, pp. 1929–1958, 2014.

[41] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in International Conference on Learning Representations (ICLR), 2015.

[42] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in International Conference on Learning Representations (ICLR), 2018.

[43] N.CarliniandD.Wagner,"Towardsevaluatingtherobustnessof neural networks," in IEEE Symposium on Security and Privacy (S&P), 2017, pp. 39–57.

[44] B. Liu and G. Apruzzese, "Attacking logo-based phishing website detectors with adversarial perturbations," arXiv preprint arXiv:2308.09392, 2023.

[45] Y. Ji, H. Wang, and D. He, "Evaluating the effectiveness and robustness of visual similarity-based phishing detection models," arXiv preprint arXiv:2405.19598, 2024.

[46] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in IEEE Security and Privacy Workshops (SPW), 2018, pp. 50–56.

[47] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, "Is BERT really robust? a strong baseline for natural language attack on text classification and entailment," in AAAI Conference on Artificial Intelligence, 2020, pp. 8018–8025.

[48] B. Geng and G. Apruzzese, "Raze to the ground: Query-efficient adversarial HTML attacks on machine-learning phishing webpage detectors," https://arxiv.org/abs/2310.03166, 2023, arXiv:2310.03166.

[49] A. Yao, W. Li, Q. Zhu, and G. Gou, "From ML to LLM: Evaluating the robustness of phishing web page detection models against adversarial attacks," arXiv preprint arXiv:2407.20361, 2024.

[50] P. Agten, W. Joosen, F. Piessens, and N. Nikiforakis, "Seven months' worth of mistakes: A longitudinal study of typosquatting abuse," in Network and Distributed System Security Symposium (NDSS), 2015.

[51] J. Spooren, D. Preuveneers, and W. Joosen, "Bypassing detection of URL-based phishing attacks using generative adversarial deep neural networks," in ACM Workshop on Artificial Intelligence and Security (AISec), 2020, pp. 53–64.

[52] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in AAAI Conference on Artificial Intelligence, 2017, pp. 2852–2858.

[53] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Advances in Neural Information Processing Systems (NeurIPS), 2014, pp. 2672–2680.

[54] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Ob ject-Oriented Software. Addison-Wesley, 1994.

[55] S. Marchal, J. Francois, R. State, and T. Engel, "PhishStorm: Detecting phishing with streaming analytics," IEEE Transactions on Network and Service Management, vol. 11, no. 4, pp. 458–471, 2014.

[56] A. Costello, "RFC 3492: Punycode: A bootstring encoding of Unicode for internationalized domain names in applications (IDNA)," IETF. https://www.rfc-editor.org/rfc/rfc3492, 2003.

[57] T. Bai, J. Luo, J. Zhao, B. Wen, and Q. Wang, “Recent advances in adversarial training for adversarial robustness,” arXiv preprint arXiv:2102.01356, 2021.
[58] Корнюхина, С. П., and О. Р. Лапонина. "Исследование возможностей алгоритмов глубокого обучения для защиты от фишинговых атак." International Journal of Open Information Technologies 11.6 (2023): 163-174.
[59] Намиот, Д. Е. Схемы атак на модели машинного обучения / Д. Е. Намиот // International Journal of Open Information Technologies. – 2023. – Т. 11, № 5. – С. 68-86. – EDN YVRDOB.
[60] Намиот, Д. Е. Осведомленность о фишинге - вопросы обучения / Д. Е. Намиот, В. А. Васенин // Современные информационные технологии и ИТ-образование. – 2025. – Т. 21, № 2. – С. 221-229. – DOI 10.25559/SITITO.021.202502.221-229. – EDN SLPOSM.

Adversarial Attacks on URL-Based Phishing Detection Models

E.D. Shustova, O.R. Laponina

Abstract – **Phishing attacks remain one of the most widespread and economically significant cyberthreats. Machine and deep learning models for URL phishing detection achieve F1 scores of 0.95–0.99 on standard datasets, but their resilience to targeted adversarial modifications of input data remains poorly understood.**

This paper proposes a systematization of phishing attack types and detection methods, a comparative analysis of URL-based architectures (Random Forest, symbolic CNN, LSTM, CNN+LSTM), a classification of adversarial attacks on phishing detectors, and a Python framework that implements seven types of black-box evasion attacks: homoglyph substitution, typosquatting, subdomain injection, URL extension, URL encoding, GAN mutation with beam search, and composite chains.

The developed framework satisfies the following requirements: generation of adversarial URLs for seven attack types; CLI with parameter groups for each type; Python API for embedding into automated pipelines; deterministic generation via a seed parameter; support for predefined and custom chains; visual highlighting of changes in verbose mode.

Experimental evaluation was performed on a dataset of 73,575 URLs (PhishTank + Marchal2014). Homoglyph modifications were shown to reduce the F1-score by 10–15% for all architectures; typosquatting by 9–14%; subdomain injection by 8% for Random Forest and 8–13% for DL models; URL extension by 10% for Random Forest and up to 4% for DL models; URL encoding by 6% for Random Forest and 0–10% for DL models (CNN+LSTM does not degrade). The full_evasion chain reduces F1 to 0.79–0.84 for all architectures; the maximum chain reduces it to 0.77–0.84. GAN mutation with heuristic scoring reduces F1 by 14–16% without access to model parameters. All attacks are transferable in black-box mode.

Practical recommendations for improving the robustness of detectors are formulated: NFKC Unicode normalization, IDN/Punycode detector, adversarial training, Random Forest and DL ensembling, and URL length limitation. The framework's throughput is up to 220,000 URLs/min for single attacks and 48,000 URLs/min for three-stage chains.

Keywords – **Phishing detection, URL, adversarial attacks, Random Forest model robustness, symbolic CNN, LSTM, CNN+LSTM, Python framework, homoglyph modifications, typosquatting, subdomain injection, URL extension, URL encoding, GAN mutation.**

REFERENCES

[1] Kaspersky Lab, "Kaspersky security bulletin 2024: Spam and spam-and-phishing-in-2024, 2024, accessed: 2025-01-10.

[2] Anti-Phishing Working Group, "Phishing activity trends report q1 2024," https://apwg.org/trendsreports/, 2024, accessed: 2025-01-15.

[3] Federal Bureau of Investigation, "Internet crime report 2023", https://www.ic3.gov/Media/PDF/AnnualReport/2023_IC3Report.pdf, 2024, accessed: 2025-01-20.

[4] IBM Security, "Cost of a data breach report 2024," https://www.ibm.com/reports/data-breach, 2024, accessed: 2025-01-20.

[5] Verizon, "2024 data breach investigations report," https://www.verizon.com/business/resources/reports/dbir/, 2024, accessed: 2025-01-15.

[6] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, "Measuring and detecting fast-flux service networks," in Network and Distributed System Security Symposium (NDSS), 2008.

[7] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," Expert Systems with Applications, vol. 117, pp. 345–357, 2019.

[8] S. Y. Yerima and M. K. Alzaylaee, "PhishGuard: A convolutional neural network based model for detecting phishing URLs with explainability analysis," arXiv preprint arXiv:2404.17960, 2024.

[9] A. Anand, S. Garg, G. Choudhary, and N. Pandey, "AntiPhishStack: LSTM-based stacked generalization model for optimized phishing URL detection," Symmetry, vol. 16, no. 2, p. 248, 2024.

[10] G.Apruzzese,P.Laskov,andA.Tastemirova,"SpacePhish:The evasion-space of adversarial attacks against phishing website detectors using machine learning," in Annual Computer Security Applications Conference (ACSAC), 2022, pp. 578–592.

[11] B. Liang, J. Su, W. Guo, Y. Shi, and X. Zhang, "Directed adversarial sampling attacks on phishing detection," Journal of Computer Security, vol. 29, no. 1, pp. 1–29, 2021.

[12] R. Liu, Y. Lin, X. Yang, J. Y. Ng, D. M. Divakaran, and J. S. Dong, "PhishIntel: Toward practical deployment of reference-based phishing detection," in ACM SIGSAC Conference on Computer and Communications Security (CCS), 2024.

[13] G. Apruzzese, M. Colajanni, L. Ferretti, and M. Marchetti, "Revisiting the performance of machine learning-based phishing URL detection," IEEE Transactions on Network and Service Management, vol. 21, no. 2, pp. 1631–1648, 2024.

[14] A.SafiandS.Singh,"Asystematicliteraturereviewonphishing website detection techniques," Journal of King Saud University – Computer and Information Sciences, vol. 35, no. 2, pp. 590– 611, 2023.

[15] G. Apruzzese, P. Laskov, and A. Tastemirova, " Multi-SpacePhish: Extending the evasion-space of adversarial attacks against phishing website detectors using machine learning," ACM Transactions on Privacy and Security, 2023.

[16] N. S. Afanasyeva, D. A. Elizarov, and T. A. Myznikova, "Klassifikatsiya fishingovykh atak i mery protivodeystviya im," Informatsionnaya bezopasnost' regionov, no. 5(89), 2022.

[17] P. Kintis, N. Miramirkhani, C. Lever, Y. Chen, R. Romero-Gomez, N. Pitropakis, N. Provos, and M. Antonakakis, "Hiding in plain sight: A longitudinal study of combosquatting abuse," in ACM SIGSAC Conference on Computer and Communications Security (CCS), 2017, pp. 569–586.

[18] T. Holgers, D. E. Watson, and S. D. Gribble, "Cutting through the confusion: A measurement study of homograph attacks," in USENIX Annual Technical Conference (ATC), 2006, pp. 261–266.

[19] M. A. Adebowale, K. T. Lwin, E. Sanchez, and M. A. Hossain, "Intelligent phishing website detection system using modified frequent pattern-growth," International Journal of Engineering and Advanced Technology, vol. 10, no. 3, pp. 22–29, 2021.

[20] L. Tang and Q. H. Mahmoud, "A survey of machine learning-based solutions for phishing website detection," Machine Learning and Knowledge Extraction, vol. 3, no. 3, pp. 672–694, 2021.

[21] G. Apruzzese, M. Colajanni, L. Ferretti, and M. Marchetti, "Evasion attacks against banking fraud detection systems," https://arxiv.org/abs/2004.06954, 2020, arXiv:2004.06954.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[23] T. Koide, D. Chiba, and M. Akiyama, "Detecting phishing sites using ChatGPT," arXiv preprint arXiv:2306.05816, 2024.

[24] S. Bell and P. Komisarczuk, "An analysis of phishing blacklists: Google Safe Browsing, OpenPhish, and PhishTank," in Australasian Computer Science Week Multiconference (ACSW), 2020.

[25] D.Sahoo,C.Liu,andS.C.Hoi,"MaliciousURLdetectionusing machine learning: A survey," arXiv preprint arXiv:1701.07179, 2017.

[26] Y. Zhang, J. Hong, and L. Cranor, "CANTINA: A content-based approach to detecting phishing web sites," in Proceedings of the 16th International Conference on World Wide Web (WWW), 2007, pp. 639–648.

[27] S. Abdelnabi, K. Krombholz, and M. Fritz, "VisualPhishNet: Zero-day phishing website detection by visual similarity," in Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS), 2020, pp. 1681–1698.

[28] Y. Lin, R. Liu, D. M. Divakaran, J. Y. Ng, Q. Z. Chan, Y. Lu, Y. Li, T. Wang, and J. S. Dong, "Phishpedia: A hybrid deep learning based approach to visually identify phishing webpages," in USENIX Security Symposium, 2021, pp. 3793–3810.

[29] T. Berners-Lee, R. Fielding, and L. Masinter, "RFC 3986: Uniform resource identifier (URI): Generic syntax," IETF. https://www.rfc-editor.org/rfc/rfc3986, 2005.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay,

"Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[31] R. S. Rao and A. R. Pais, "Detection of phishing websites using an efficient feature-based machine learning framework," Neural Computing and Applications, vol. 31, no. 8, pp. 3851–3873, 2019.
[32] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785–794.
[33] Y. Wang, J. Zhang, and H. Li, "Comprehensive evaluation of adversarial perturbations against ML-based ethereum phishing detection systems," ACM Transactions on the Web, 2025.
[34] P. Yang, G. Zhao, and P. Zeng, "Phishing website detection based on multidimensional features driven by deep learning," IEEE Access, vol. 7, pp. 15 196–15 209, 2019.
[35] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in Advances in Neural Information Processing Systems (NeurIPS), vol. 28, 2015, pp. 649–657.
[36] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
[37] A. S. Bozkir, F. C. Dalgic, and M. Aydos, "GramBeddings: A new neural network for URL-based malicious web site detection through n-gram embeddings," Computers & Security, vol. 124, p. 103016, 2023.
[38] H. Le, Q. Pham, D. Sahoo, and S. C. Hoi, "URLNet: Learning a URL representation with deep learning for malicious URL detection," https://arxiv.org/abs/1802.03162, 2018, arXiv:1802.03162.
[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in International Conference on Learning Representations (ICLR), 2015.
[40] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, vol. 15, pp. 1929–1958, 2014.
[41] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in International Conference on Learning Representations (ICLR), 2015.
[42] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in International Conference on Learning Representations (ICLR), 2018.
[43] N.Carlini and, D.Wagner,"Towards evaluating hero bustness of neural networks," in IEEE Symposium on Security and Privacy (S&P), 2017, pp. 39–57.
[44] B. Liu and G. Apruzzese, "Attacking logo-based phishing website detectors with adversarial perturbations," arXiv preprint arXiv:2308.09392, 2023.
[45] Y. Ji, H. Wang, and D. He, "Evaluating the effectiveness and robustness of visual similarity-based phishing detection models," arXiv preprint arXiv:2405.19598, 2024.
[46] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in IEEE Security and Privacy Workshops (SPW), 2018, pp. 50–56.
[47] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, "Is BERT really robust? a strong baseline for natural language attack on text classification and entailment," in AAAI Conference on Artificial Intelligence, 2020, pp. 8018–8025.
[48] B. Geng and G. Apruzzese, "Raze to the ground: Query-efficient adversarial HTML attacks on machine-learning phishing webpage detectors," https://arxiv.org/abs/2310.03166, 2023, arXiv:2310.03166.
[49] A. Yao, W. Li, Q. Zhu, and G. Gou, "From ML to LLM: Evaluating the robustness of phishing web page detection models against adversarial attacks," arXiv preprint arXiv:2407.20361, 2024.
[50] P. Agten, W. Joosen, F. Piessens, and N. Nikiforakis, "Seven months' worth of mistakes: A longitudinal study of typosquatting abuse," in Network and Distributed System Security Symposium (NDSS), 2015.
[51] J. Spooren, D. Preuveneers, and W. Joosen, "Bypassing detection of URL-based phishing attacks using generative adversarial deep neural networks," in ACM Workshop on Artificial Intelligence and Security (AISec), 2020, pp. 53–64.
[52] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence generative adversarial nets with policy gradient," in AAAI Conference on Artificial Intelligence, 2017, pp. 2852–2858.
[53] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Advances in Neural Information Processing Systems (NeurIPS), 2014, pp. 2672–2680.
[54] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Ob ject-Oriented Software. Addison-Wesley, 1994.
[55] S. Marchal, J. Francois, R. State, and T. Engel, "PhishStorm: Detecting phishing with streaming analytics," IEEE Transactions on Network and Service Management, vol. 11, no. 4, pp. 458–471, 2014.
[56] A. Costello, "RFC 3492: Punycode: A bootstring encoding of Unicode for internationalized domain names in applications (IDNA)," IETF. https://www.rfc-editor.org/rfc/rfc3492, 2003.
[57] T. Bai, J. Luo, J. Zhao, B. Wen, and Q. Wang, "Recent advances in adversarial training for adversarial robustness," arXiv preprint arXiv:2102.01356, 2021.
[58] Kornjuhina, S. P., and O. R. Laponina. "Issledovanie vozmozhnostej algoritmov glubokogo obuchenija dlja zashhity ot fishingovyh atak." International Journal of Open Information Technologies 11.6 (2023): 163-174.
[59] Namiot, D. E. Shemy atak na modeli mashinnogo obuchenija / D. E. Namiot // International Journal of Open Information Technologies. – 2023. – T. 11, # 5. – S. 68-86. – EDN YVRDOB.
[60] Namiot, D. E. Osvedomlennost' o fishinge - voprosy obuchenija / D. E. Namiot, V. A. Vasenin // Sovremennye informacionnye tehnologii i IT-obrazovanie. – 2025. – T. 21, # 2. – S. 221-229. – DOI 10.25559/SITITO.021.202502.221-229. – EDN SLPOSM.