Phish-Hook: Detecting Phishing Certificates Using Certificate Transparency Logs

Edona Fasllija[1], Hasan Ferit Enişer[2], and Bernd Prünster[3]

[1] A-SIT Secure Information Technology Center Austria, Graz, Austria
edona.fasllija@a-sit.at
[2] Computer Engineering Department, Bogazici University, Istanbul, Turkey
hasan.eniser@boun.edu.tr
[3] Institute of Applied Information Processing and Communications, Graz University of Technology, Graz, Austria bernd.pruenster@iaik.tugraz.at

Abstract. Certificate misissuance is a growing issue in the context of phishing attacks, as it leads inexperienced users to further trust fraudulent websites, if they are equipped with a technically valid certificate. *Certificate Transparency* (CT) aims at increasing the visibility of such malicious actions by requiring *certificate authorities* (CAs) to log every certificate they issue in public, tamper-proof, append-only logs. This work introduces *Phish-Hook*, a novel approach towards detecting phishing websites based on machine learning. Phish-Hook analyses certificates submitted to the CT system based on a conceptually simple, well-understood classification mechanism to effectively attest the phishing likelihood of newly issued certificates. Phish-Hook relies solely on CT log data and foregoes intricate analyses of websites' source code and traffic. As a consequence, we are able to provide classification results in near real-time and in a resource-efficient way. Our approach advances the state of the art by classifying websites according to five different incremental certificate risk labels, instead of assigning a binary label. Evaluation results demonstrate the effectiveness of our approach, achieving a success rate of over 90%, while requiring fewer, less complex input data, and delivering results in near real-time.

Keywords: TLS · Certificate Transparency · CA · certificate misissuance · machine learning · phishing detection

1 Introduction

Transport Layer Security (TLS) [18], critically relies on *certificate authorities* (CAs) as trust anchors. A series of security incidents related to either compromised CAs or poor CA certification practices have shown that this high degree of trust put into certificate authorities was, at times, misplaced. A prominent example is the incident related to the compromised Dutch CA *DigiNotar* [10] where attackers managed to issue TLS certificates for fake websites impersonating Gmail and Facebook. Similar incidents occurred with a Malaysian subordinate CA *DigiCert Sdn. Bhd.* and the large U.S.-based CA *TrustWave* [13]. Events like

these challenge the conceptually simple trust model the current TLS *public key infrastructure* (PKI) system is based on.

Moreover, the popularity of free and automated TLS certificates by companies like *Let's Encrypt* and *Cloudflare* has led to a massive surge in the use of automatically-issued certificates on phishing sites. A recent statistical report from *PhishLab* [23] indicates that 49% of phishing sites were using HTTPS in the third quarter of 2018. This percentage has rapidly increased from 25% just one year ago, and from 35% in the second quarter of 2018. Current security mechanisms in browsers fail at detecting fraudulent websites if they are provisioned with mistakenly or maliciously issued certificates that are technically valid. Furthermore, when such a misissuance happens, it can take weeks or even months until the suspect certificates are detected and revoked. This window of vulnerability gives malicious actors plenty of time to do damage.

Google responded to the need for auditing the web's PKI system by implementing *Certificate Transparency* (CT) [13] — an open and public framework that audits and monitors TLS certificates in near real-time. CT publicly records TLS certificates in append-only logs as they are issued, in a way that enables anyone to audit a CA's activity and notice the issuance of suspect certificates for the domains they own. The instant visibility of newly issued certificates can significantly reduce the amount of time needed until a malicious site or CA misconduct can be detected.

Our main research contribution is an effective mechanism that detects phishing websites in near real-time by leveraging machine learning techniques to evaluate CT logs. We identify eight features and build a classification model that is able to utilize different algorithms to maximize fraud detection rates. This model is trained and evaluated on real data and accounts for the asymmetric distribution between legitimate websites and phishing sites for broad real-world applicability.

This paper is organized as follows: Section 2 summarizes the fundamental principles of the web's PKI, TLS certificates, Certificate Transparency, and covers phishing techniques. Section 3 describes previous research done on the topic. Section 4 delves into the design and the properties of *Phish-Hook*, our machine-learning-based phishing detection approach. Section 5 focuses on the evaluation of our approach and reports on our model's performance based on different classification models. Section 6 finally concludes this work and elaborates on possible future work directions.

2 Background

Confidentiality and integrity of web traffic are ensured using the HTTPS protocol and the supporting public key infrastructure. During the setup of a TLS channel and its underlying TCP connection, a client is required to authenticate a server by validating its certificate. A website is successfully authenticated if it uses a certificate that has not expired (and has not been revoked) and if a certificate chain can be built up to one of the certificate authorities present in the browser's trusted CA list.

The web's current PKI system allows *any* trusted CA, or intermediate CA, to issue certificates for *any* subject identity. This assumption of trustworthy CAs introduces a vulnerability to attacks based on improperly issued certificates, either as a result of CA compromise, negligence, errors, or even malicious behavior. The following section explains how this can be exploited to enable phishing, while Section 2.2 explains the reasoning behind utilizing Certificate Transparency as a promising building block in the fight against fraud.

2.1 Phishing Attacks

Malicious actors use several ways to trick users into visiting a website with a domain name similar to that of a legitimate website. Examples include typo-squatting [21], homoglyph (name spoofing) attacks [5], or incorporating a legitimate domain as a prefix, inner part, or suffix, as explained below.

Considering the domain `phish-hook.com`, a typo-squatting attack would try to register domains by incorrectly spelling the domain name such as `phihs-hook.com` or `phish-hok.com`. On the other hand, homoglyphic attacks rely on character substitution using look-alike glyphs from the Unicode sets to create fake domain names that are nearly indistinguishable from real ones to the naked eye. A quick look at the *confusables* file [6] published by the Unicode consortium, reveals that just for the character `i` in phish-hook, up to 41 look-alike glyphs exist that can be utilized by attackers to produce examples such as `phish-hook.com`, `phish-hook.com`, or `ph`*`i`*`sh-hook.com`.

In addition, domains such as `www-phish-hook.com`, `login-phish-hook.com`, and `www.phish-hook.com.malicious.fakedomain.name` can be built by incorporating the legitimate domain name into a longer domain. More details on these techniques are provided in Section 4.2, where we explain how features were extracted from the certificates present in CT logs as part of our solution. The following section provides an overview of Certificate Transparency and the motivation behind the system's design.

2.2 Certificate Transparency

Certificate Transparency (CT) provides visibility of newly issued certificates and CA operations. This open framework monitors and audits TLS certificates by complementing the TLS ecosystem with three main components: *Certificate Logs*, *Monitors*, and *Auditors*. Certificate log servers maintain cryptographically assured, append-only logs of issued certificates. Monitor servers periodically check these logs to determine whether an illegitimate certificate has been issued for a particular domain. Auditors check whether logs are cryptographically consistent and whether a particular certificate is registered in the logs.

The strength of the framework stems from the append-only, cryptographically-assured nature of the logs. On a technical level, this is accomplished by relying on a *Merkle Tree* (a data structure made up of linked cryptographic hashes [14]). This ensures that back-dated certificates cannot be inserted into the log, and added certificates cannot be edited or deleted afterward. In the CT-augmented

certificate issuance process, a CA submits a newly-issued certificate to the log, and is provided with a *Signed Certificate Timestamp* (SCT)—a proof of inclusion to the log— in return. The SCT can be added to a certificate either as an X.509v3 certificate extension, a TLS extension or during the handshake as part of OCSP stapling [13]. This enables clients to verify whether the SCT was provided by a trusted CT log by validating its signature and eventually deciding on whether to accept a certificate as valid or not according to their CT policy.

The following section elaborates on how machine learning has been used in the past to detect fraudulent websites. It summarises both work conducted prior to the introduction of CT and works already leveraging CT.

3 Related Work

One of the earliest proposals for using machine learning to detect phishing websites based on certificates was published by Mishari et al. [15]. The authors crawled certificates of both legitimate and phishing websites and used them to build a classifier based on, amongst others, features either directly extracted or computed from certain X.509 certificate fields. Their proposal was the first to showcase the potential of using certificate information beyond client-side server authentication to identify fraudulent websites that use HTTPS.

In another more recent effort, Dong et al. [8] propose to employ *Deep Neural Networks* (DNNs) to identify potentially rogue certificates (and eventually CA misconduct) in a timely manner using features extracted from standard X.509 certificates. The authors additionally address the dataset imbalance (between rogue vs. benign certificate sets) by generating artificial rogue certificates.

Similar to the work by Dong et al. [8], Torroledo et al. [22] propose to use DNNs to detect malicious use of TLS certificates. The authors perform a detailed feature engineering and identify more than thirty features to classify malicious websites. Results indicate that their system can detect malware certificates and phishing certificates with an accuracy of a 94.8% and 88.6%, respectively.

Ghafir et al. [9] investigate the problem of malicious certificate detection as means to defend against *advanced persistent threats* (APTs). They base their proposal to detect APT *command and control* (C&C) communications on a blacklist of three certificate fields, namely SHA-1 fingerprints, serial and subject. The module they propose analyses network traffic, filters secure communications and matches certificates that were used in these communications to the certificates in the blacklist.

Kumar et al. [12] construct a certificate linter that checks certificates in the wild for compliance to the *CA/Browser Forum Baseline Requirements* [3] and RFC5280 [7]. Their findings indicate that the certificate misissuance rate has dramatically dropped since 2017 (down to 0.02%). Nonetheless, authors propose an alternative way to make CA operations auditable and showcase how their linter can be harnessed to identify poor CA practices.

It was not until very recently that CT was envisioned as a potential data source for the detection of suspicious phishing domains. In 2018, companies like

Facebook[4] and SSL Mate's *Certspotter*[5] started offering notification services about suspiciously issued certificates to subscribing domain owners based on CT log monitoring. Nonetheless, their approaches to solving the phishing detection problem are not disclosed nor analysed for efficacy or accuracy.

Finally, Scheitle et al. [19] conducted a pilot experiment in order to gain insights about the viability of using CT logs as phishing detection source. The authors employ regular expression matching to find potential phishing domains by means of CT. They observe that most phishing domains are constructed by combining fully qualified domains (FQDN) of popular legitimate target domains. Their paper aims at investigating variant implications that CT has had on the Internet ecosystem and is not focused solely on phishing detection. They do not employ any automated machine-learning based detection mechanism, but rather base their phishing detection approach on regular expression matching and visual inspection. They conclude that CT being used as data source for phishing detection opens a promising new research direction.

We bring this idea forward by using CT logs as source for valuable data to train and validate a classifier that predicts the phishing likelihood of certificates submitted to the logs. We automate the phishing detection process by utilising machine learning algorithms and employing a heuristics based scoring methodology to assign five different phishing scores. In contrast to other machine-learning-based approaches summarized in this section, we do not require downloading a large set of certificates, but collect and label the data needed while streaming certificate updates from the CT logs. By applying our classifier to certificates newly submitted to the CT logs, we can detect phishing attempts in near real time and dramatically reduce the window of vulnerability for such attacks.

4 Phish-Hook

We propose the idea of using CT logs as the sole data source for phishing detection by presenting a machine-learning-based solution called *Phish-Hook*. Compared to other contributions discussed in Section 3, we do not require to download and parse the certificates of corresponding websites, nor do we use extra features from the websites' source code or monitor traffic. As we will show in Section 5, this approach delivers highly accurate results based on directly applying machine learning techniques to certificates. This phishing detection system is composed of three main components: namely the *Certificate Collector*, the *Feature Extractor*, and the *Classifier*. The CT logs feed the *Certificate Collector*, which in turn passes the parsed CT logs to the *Feature Extractor* component. The set of attributes generated from the *Feature Extractor* is finally used to train our *Classifier* model. New certificates streamed from the CT logs are then fed into the trained *Classifier* to be classified into on of five incremental phishing likelihood scores. Figure 1 provides an illustration of Phish-Hook's main components.

[4] `https://developers.facebook.com/tools/ct/subscriptions/`
[5] `https://sslmate.com/certspotter/`

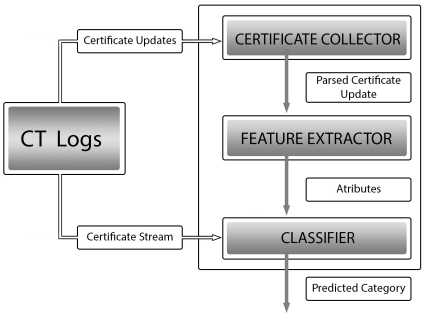

Fig. 1. Phish-Hook System Components

The following section describes the data collection methodology, while Section 4.2 elaborates on the extraction of certificate features, and Section 4.4 discusses the algorithms used to train the *Classifier*'s model.

4.1 Data Collection

Given our aspiration of directly leveraging the CT logs to build the classifier model, we built our own training dataset and used the *CertStream*[6] open-source library to interact with the CT network and aggregate CT log data. The Structure of the parsed CT logs is shown in Listing 1:

Listing 1. Parsed Certificate Log Update Entry

```
{"message_type": "certificate_update",
"data": {
      "update_type": "X509LogEntry",
            "leaf_cert": {...
                  "subject": {
                        "aggregated": "/CN=phish-hook.com",
                        "C": null,
                        "ST": null,
                        "L": null,
                        "O": null,
                        "OU": null,
                        "CN": "phish-hook.com"
                   },
            ...
            "all_domains":
            [
             "login.phish-hook.com"
             "phish-hook.com"]
            },
            "chain": [{
                  "subject": {
                        "aggregated": "/C=US/O=Let's Encrypt/CN=Let's Encrypt Authority X3",
                        "C": "US",
                        "ST": null,
                        "L": null,
                        "O": "Let's Encrypt",
                        "OU": null,
                        "CN": "Let's Encrypt Authority X3"},
                        ...],
            "cert_index": 27910635, }}}
```

The following section provides details on the features used to train our classifier model extracted from certificate updates submitted to CT logs.

[6] https://medium.com/cali-dog-security/introducing-certstream-3fc13bb98067

4.2 Feature Selection

Certificate Transparency augments raw certificate data with time as another dimension. Our contribution is rooted in the assumption that small amounts of highly characteristic data are meaningful enough to enable classification of phishing sites based on a relatively simple machine learning model. In contrast to related work on this sector, our solution thus employs only a comparatively small set of features and still delivers highly accurate results. In addition, PhishHook evaluates features directly from the parsed CT log entries without requiring to download the respective certificates. On a technical level, the data points were extracted from the CT log entry fields representing domain names (`[data][leaf_cert][subject][aggregated]` and `[data][all_domains]`) and the certificate issuer fields (`[chain][subject][aggregated]`) of each certificate update entry.

As existing data indicating which features are relevant when it comes to detecting phishing websites from CT log entries is hard to come by, we aggregated and analyzed reports on phishing in general. Based on this, we derived the following feature set:

F1 small_levenshtein_distance Section 2 summarized some of the most common techniques utilised by attackers to generate misleading domain names for phishing attacks, such as typosquatting, homograph attacks, etc.. Our first feature is based on this, and on further observations made by Scheitle et al. [19] suggesting that the majority of phishing website certificates registered in the CT logs is constructed by incorporating domain names of popular legitimate domains. The value of F1 is assigned by calculating the Levenshtein Distance — a measure of similarity between two strings — of sub-words of the domain registered with the certificate to suspicious popular keywords (for example: `phish-hook` vs `phish_hook`). Table 4.2 summarizes the popular keywords that were used in order to calculate F1. In case the computed distance to the keywords is below a certain threshold, we consider this an indicator of suspiciousness.

F2 deeply_nested_subdomains We consider domain names with unusually long subdomains such as `www.phish-hook.com.security.account-update.gq` to be an indicator of suspiciousness. Similar domains have been widely used by attackers to impersonate legitimate websites by hiding the primary domain in deeply nested subdomains. These attacks particularly target small devices such as tablets or mobile phones that can not display the full (long) domain at once.

F3 issued_from_free_CA Section 1 already pointed out that HTTPS phishing has been increasing significantly in the past couple of years and is about to become prevalent. The ubiquity of automated, fast, and free certificates has given both good and bad actors the advantage of easily obtaining a SSL/TLS certificate for their websites. The problem actually does not lie on the free and automated certificate issuance itself, but on the ongoing debate on which actors of the Internet ecosystem have the responsibility of policing the content of nature

of websites. In one of their position papers [1], Let's Encrypt disagrees that it is a CA's responsibility to check for malicious or phishing content at the level of domain validated certificates. They instead prefer to delegate this responsibility to services such as Google Safe Browsing or Microsoft SmartScreen. Additional reports [16] ranking such free CAs in top positions with respect to the number of phishing certificates blocked led us to consider certificates obtained from free CAs as a potential indicator of suspiciousness.

F4 suspicious_ tld Unlike the other lower level domains that can be generally reserved by domain owners, top-level domains are generally prominent domains such as `.com`,`.net`, `.edu` or `.org` that end users are familiar with. Malicious actors often target these top-level domains in their attempt to create malicious sites. The low cost at which a large number of newly added TLDs is available makes certain TLDs more popular amongst attackers. Based on observations made from available reports [20] on most abused TLDs, we consider top-level domains such as `.ga', '.gdn','.bid''.country','.kim'`, etc. that were widely adopted for phishing purposes as suspicious. The complete list of the TLDs considered can be found in Table 4.2.

F5 inner_ tld_ in_ subdomain Attackers may include popular top-level domains (such as 'org', 'com', or 'net') in the inner domain in order to mislead users that are familiar with them into trusting a fraudulent website. The presence of such a TLD in an inner sub-domain is therefore considered suspicious.

F6 suspicious_ keywords Another well-known phishing technique is the inclusion of popular keywords from famous applications of social media, commerce, or cryptocurrency in a domain name. We therefore check whether each CT certificate update entry contains one of the keywords present in Table 4.2, and consider a match suspicious.

F7 high_ shannon_ entropy F7 aims to detect algorithmically generated malicious domain names in particular. This feature lays its foundation on the observation that these domain names differ significantly in terms of randomness when compared to human generated domains. In order to do so, we calculate the Shannon entropy — i.e. degree of randomness — of the domain a certificate was issued for. An unusually high entropy may then serve as indicator for maliciously issued certificates from attackers.

F8 hyphens_ in_ subdomain F8 is similar to F2, but instead of checking for the presence of multiple periods ('.'), we check for the presence of multiple hyphens ('-') in the sub-domain, as both of these characters can be used to attach popular keywords of legitimate domains to generate malicious ones. Hence, for F8, we consider an unusually high number of hyphens an indicator of a suspicious website.

Table 1. Suspicious keywords and TLDs [24]

Generic		Apple	Email	Cryptocurrency	Social Media	Financial	E-commerce	Other	TLDs		Misc.
activity	office	appleid	outlook	poloniex	facebook	moneygram	overstock	skype	'.bank'	'.online'	.com-
alert	online	icloud	office365	coinhive	tumblr	westernunion	alibaba	github	'.business'	'.party'	-com.
purchase	recover	iforgot	microsoft	bithumb	reddit	bankofamerica	aliexpress		'.cc'	'.pw'	.net-
authentication	safe	itunes	windows	kraken	youtube	wellsfargo	leboncoin		'.center'	'.racing'	.org-
authorize	secure	apple	protonmail	localbitcoin	twitter	paypal	amazon	netflix	'.cf'	'.ren'	cgi-bin
bill	security		tutanota	bitstamp	linkedin	citigroup			'.click'	'.review'	.com-
client	service		hotmail	bittrex	instagram	santander			'.club'	'.science'	.net.
support	transaction		gmail	blockchain	flickr	morganstanley			'.country'	'.stream'	.org.
unlock	update		google	bitflyer	whatsapp	barclays			'.download'	'.study'	.com.
wallet	account		outlook	coinbase		hsbc			'.ga'	'.support'	.gov-
form	login		yahoo	hitbtc		scottrade			'.gb'	'.tech'	.gov.
log-in	password,		google	lakebtc		ameritrade			'.gdn'	'.tk'	.gouv-
live	signin		yandex	bitfinex		merilledge			'.gq'	'.top'	-gouv-
manage	sign-in			bitconnect		bank			'.info'	'.vip'	.gouv.
verification	verify			coinsbank					'.kim'	'.win'	
webscr	invoice								'.loan'	'.work'	
authenticate	confirm								'.men'	'.xin'	
credential	customer								'.ml'	'.xyz'	
									'.mom'		

4.3 Classification Workflow

In a nutshell, our system works as follows: We stream certificate updates from CT logs, while simultaneously labelling the data for each feature. We also employ a heuristic methodology to compute a total phishing likelihood score according to the presence or absence of a feature, or the computed value of a feature. We use this overall score to classify the certificate and assign the resulting feature called ***phishing_likelihood_category*** out of five different categories, namely **legitimate**, **potential**, **likely**, **suspicious**, and **highly-suspicious**.

4.4 Learning Phase

This section goes into the details of the training phase. After collecting and labelling the data from CT, we employ supervised learning algorithms to train several classifier models in order to predict a certificate's phishing likelihood.

One challenge was having *imbalanced* classes: The number of phishing websites recorded publicly in CT logs is much smaller than the number of legitimate websites. Thus, collecting the dataset from CT logs results in a very small number of datapoints labelled as *phishing* compared to the number of datapoints labelled as *legitimate*. To give an idea of the imbalance in the data: in a dataset of approximately 600000 datapoints, the number of *highly-suspicious, suspicious, likely, potential* and *legitimate* instances are 223, 230, 251, 719 and 602676, respectively. This situation is referred to as *imbalanced classes problem* [11] and is a common phenomenon in phishing detection-related learning processes.

Oversampling or undersampling techniques are potential countermeasures to overcome this challenge. We employ SMOTE (Synthetic Minority Over-sampling Technique)[4] and random *undersampling* to address the challenge presented by the imbalanced classes problem. We chose to oversample *highly-suspicious, suspicious, likely and potential* classes and undersample *legitimate* instances, resulting with a much more balanced dataset.

5 Evaluation

This section describes how Phish-Hook was evaluated and discusses the obtained results. It compares different key performance measures of our system against existing work and clearly illustrates that our work presents a significant step forward towards phishing detection. Most importantly, our system achieves its results in near real-time based on a small set of only eight features and a simple machine learning model. This not only makes it possible to react to phishing sites as soon as they emerge, but also enables debugging and presenting the decision-making process in a humanly-comprehensible manner. As our system relies on traditional machine learning approaches, training and classification outperform Deep Neural Networks in the time domain.

The following section describes the test setup and how we trained and evaluated our model based on an existing, pre-classified data set. Section 5.2 then establishes which metrics we used to measure the performance of Phish-Hook, while Section 5.3 pits different classifiers against each other to evaluate which one is best suited for the task at hand.

5.1 Training Dataset

To evaluate the performance of Phish-Hook, we made use of the pre-classified phishing detection dataset publicly available under the UCI Machine Learning Repository [2]. This dataset consists of 11055 data points with 30 features. Part of the features correspond directly to X.509 certificate fields, while others are derived from certificates fields and/or website source code. Each feature takes a ternary value of [-1,0,1] representing *phishing*, *suspicious*, and *legitimate* respectively. Unlike features, result labels can take only two values: *phishing* and *legitimate*.

This data set and the classification process do not account for CT log data. However, our small set of simple features can be modelled as a subset of it and thus aligns well with the pre-classified data to provide a ground truth. This works because CT log data overlaps with the features present in the UCI Machine Learning Repository dataset. The following section briefly presents the metrics used for the evaluation process.

5.2 Metrics

In machine learning, classification is defined as the problem of assigning a new observation to one category, based on a training set of data containing observations (or instances) whose category membership is known. In our case, we basically want to predict if a website is legitimate or not.

The most basic metric used to quantify the performance of a classifier is *accuracy*, i.e. the ratio of the number of correct predictions to the total predictions made. In problems with (highly) imbalanced classes, however, accuracy can easily be boosted by always outputting the category of the largest class. Consequently, accuracy is inadequate on its own as a performance measure and additional

metrics such as precision, recall and the so-called *F_1 score* (see Equation 3) are typically used to measure the true aptitude of a classifier.

Precision and Recall are calculated according to Equations 1 and 2, with tp, fp, and fn denoting the number of correctly identified instances, the number of incorrectly identified instances, and the number of incorrectly rejected instances respectively:

$$Precision = \frac{tp}{tp + fp} \tag{1}$$

$$Recall = \frac{tp}{tp + fn} \tag{2}$$

$$F_1\ score = \frac{2 \times Recall \times Precision}{Recall + Precision} \tag{3}$$

All of these scores apply to binary classification problems, which matches the UCI Machine Learning Repository dataset. Our solution, on the other hand, introduces a granular metric, classifying websites in one of five categories in the range of [**legitimate**, **potential**, **likely**, **suspicious**, **highly-suspicious**]. In order to evaluate our results against the pre-classified data, we introduced **suspicious** as the threshold for classifying a certificate to be issued for a phishing website. The following section presents the classification performance of Phish-Hook based on different classifiers.

5.3 Results

Although Deep Neural Networks are currently hailed as an almost universal solution (not only) to classification problems, we have intentionally focused our work on classical machine learning approaches for reasons of performance and comprehensibility. We thus pitched the performance of well-understood algorithms such as k-nearest neighbour (KNN), support vector machines (SVMs), decision tree classifiers (DT), and multilayer perceptrons (MLP) against each other.

We report accuracy, precision, recall, and F_1 scores regarding the classification of phishing websites for each approach in Table 2. On a technical level, our work is based on the *scikit-learn* [17] Python library.
Evaluation results are reported for various parameters tuned for each algorithm, such as maximum depth for DT, network size for MLP, and the penalty parameter C for the SVM classifier. The results demonstrate the effectiveness of our approach, with an accuracy of over 90%, while maintaining precision, recall, and F_1 scores of also over 90%. Support vector classifiers outperform others for the certificate classification task, closely followed by decision trees by a small margin. Reported results show that our approach can outperform existing solutions while requiring fewer, less complex features. Labelling happens on-the-fly without the need to download immense amounts of certificate data and results in accurately detecting phishing attempts in almost real time when applied to newly submitted CT log data.

Table 2. Classification results

	Parameters	Accuracy	Precision	Recall	F_1 score
DT	max_depth=2	91.06	91.12	91.06	91.07
	max_depth=5	91.42	91.46	91.42	91.39
	max_depth=10	89.39	89.44	89.39	89.40
SVM	kernel = 'linear' C = 0.03	91.62	91.68	91.62	91.58
	kernel = 'linear' C = 0.3	91.29	91.37	91.29	91.25
	kernel = 'linear' C = 1	91.39	91.45	91.39	91.35
KNN	k=1	86.41	86.40	86.41	86.37
	k=3	86.38	86.40	86.38	86.32
	k=10	87.23	87.23	87.23	87.19
MLP	network_size=3×5	90.08	90.16	90.08	90.03
	network_size=5×10	89.55	89.94	89.55	89.44
	network_size=1×100	89.06	89.63	89.06	88.92

5.4 Discussion

The first major insight from these results is that our assumption about the choice of features to extract from CT log entries to identify phishing sites actually holds true for real-world data. Most importantly, this confirms that Certificate Transparency log data is indeed a valuable source of data that can be machine-processed to mount automated alert systems. In addition, using traditional, well-understood machine learning techniques results in a system whose classification process is comprehensible by humans and thus debuggable. We therefore argue for the use of simple machine learning models such as SVMs and DTs for two reasons: Firstly, the inner workings of the models are the easiest to understand and align with human intuition and basic algorithmic though processes. Secondly, classifers such as decision trees consume the least resources both during training and classification and can therefore be operated on commodity hardware. By solely relying on CT log data, the network traffic produced by Phish-Hook is also kept to a minimum. Our system's low demand for computational resources makes it even possible to deploy it on end-user devices such that it can alert users whenever they are about to access a phishing website. In addition, we advance the current state of the art with respect to detecting fraudulent websites fitted with genuine certificates by providing more than just an absolute (binary) decision, about a website's legitimacy. This aligns with the transparent decision making process in the sense that uncertainty is reflected in the classification results, whenever it arises.

6 Conclusions

This work presented *Phish-Hook*, an effective and accurate CT-log-based phishing detection system using classical machine learning algorithms. By not relying on deep neural networks, our system cannot only be trained efficiently, but remains debuggable and humanly comprehensible while in operation. As phishing heavily relies on the human factor to be successful, we firmly believe that the same holds true for phishing detection systems. We advance the current state of the art in phishing detection not only in a purely technical manner but also by our process being transparent to the user, providing more granular classification results according to five different incremental certificate risk labels.

On a technical level, our design is based on the assumption that a small set of eight features extracted directly from CT log data is sufficient to successfully classify phishing websites. Thus, Phish-Hook foregoes the need to analyze website source codes or inspect traffic. Evaluation results show that this assumption holds true, as our system outperforms existing solutions and is able to correctly identify more than 90% of phishing websites in near real-time. Our approach thus demonstrates the utility of decision trees and support vector machines—classical machine learning algorithms—for the problem at hand. This presents a major advantage over the likes of deep learning, as not only the results, but also the process of obtaining them remains intelligible. As a consequence, Phish-Hook can be improved and extended in intuitive, straight forward ways and the results will always be comprehensible by humans. Potential future work directions thus include the incorporation of additional features extracted from other CT log fields, such as validity period, extensions, etc.

In summary, Phish-Hook is able to reliably classify phishing websites based solely on CT log data in near real-time as they appear. This can significantly reduce the time it takes to detect phishing websites and consequently mitigate their impact.

References

1. Aas, J.: The ca's role in fighting phishing and malware. `https://letsencrypt.org/2015/10/29/phishing-and-malware.html`, accessed: 2019-04-29
2. Asuncion, A., Newman, D.: Uci machine learning repository (2007)
3. Ca/browser forum baseline requirements documents. `https://cabforum.org/baseline-requirements-documents/`, accessed: 2019-04-13
4. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: Synthetic minority over-sampling technique. J. Artif. Int. Res. **16**(1), 321–357 (Jun 2002), `http://dl.acm.org/citation.cfm?id=1622407.1622416`
5. Homoglyph advanced phishing attacks. `https://www.cisco.com/c/en/us/support/docs/security/email-security-appliance/200146-Homoglyph-Advanced-Phishing-Attacks.pdf`, accessed: 2019-04-13
6. Consortium, U.: Recommended confusable mapping for IDN (2015), `https://www.unicode.org/Public/security/8.0.0/confusables.txt`, accessed: April 13, 2019

7. Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., Polk, W.: Rfc 5280: Internet x. 509 public key infrastructure certificate and certificate revocation list (crl) profile. IETF, May (2008)
8. Dong, Z., Kane, K., Camp, L.J.: Detection of rogue certificates from trusted certificate authorities using deep neural networks. ACM Transactions on Privacy and Security (TOPS) **19**(2), 5 (2016)
9. Ghafir, I., Prenosil, V., Hammoudeh, M., Han, L., Raza, U.: gmalicious ssl certificate detection: A step towards advanced persistent threat defence. In: Proceedings of the International Conference on Future Networks and Distributed Systems. p. 27. ACM (2017)
10. Hoogstraaten, H.: Black tulip report of the investigation into the diginotar certificate authority breach (08 2012)
11. Kotsiantis, S., Kanellopoulos, D., Pintelas, P., et al.: Handling imbalanced datasets: A review. GESTS International Transactions on Computer Science and Engineering **30**(1), 25–36 (2006)
12. Kumar, D., Wang, Z., Hyder, M., Dickinson, J., Beck, G., Adrian, D., Mason, J., Durumeric, Z., Halderman, J.A., Bailey, M.: Tracking certificate misissuance in the wild. In: 2018 IEEE Symposium on Security and Privacy (SP). pp. 785–798. IEEE (2018)
13. Laurie, B., Langley, A., Kasper, E.: Certificate transparency. Tech. rep. (2013)
14. Merkle, R.C.: A Digital Signature Based on a Conventional Encryption Function. In: Advances in Cryptology — CRYPTO '87. pp. 369–378. Lecture Notes in Computer Science, Springer (1987)
15. Mishari, M.A., De Cristofaro, E., Defrawy, K.E., Tsudik, G.: Harvesting ssl certificate data to identify web-fraud. arXiv preprint arXiv:0909.3688 (2009)
16. Phishiest certificate authorities. `https://toolbar.netcraft.com/stats/certificate_authorities`, accessed: 2019-04-29
17. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
18. Rescorla, E., Dierks, T.: The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Aug 2008). https://doi.org/10.17487/RFC5246, `https://rfc-editor.org/rfc/rfc5246.txt`
19. Scheitle, Q., Gasser, O., Nolte, T., Amann, J., Brent, L., Carle, G., Holz, R., Schmidt, T.C., Wählisch, M.: The rise of certificate transparency and its implications on the internet ecosystem. In: Proceedings of the Internet Measurement Conference 2018. pp. 343–349. ACM (2018)
20. Spamhaus: The 10 most abused top level domains. `https://www.spamhaus.org/statistics/tlds/`, accessed : 2019-04-30
21. Szurdi, J., Kocso, B., Cseh, G., Spring, J., Felegyhazi, M., Kanich, C.: The long "taile" of typosquatting domain names. In: 23rd {USENIX} Security Symposium ({USENIX} Security 14). pp. 191–206 (2014)
22. Torroledo, I., Camacho, L.D., Bahnsen, A.C.: Hunting malicious tls certificates with deep neural networks. In: Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security. pp. 64–73. ACM (2018)
23. Volkman, E.: 49 percent of phishing sites now use https. Tech. rep. (2018), `https://info.phishlabs.com/blog/49-percent-of-phishing-sites-now-use-https`
24. x0rz: Phishing catcher. `https://github.com/x0rz/phishing_catcher`