**NOTES (**and SUMMARY)

**C*atching Transparent Phish: Analyzing and Detecting MITM Phishing*** *Toolkits* – Stonybrook

**“[sic]” means that the words are taken directly from the source, as it were.**

**(** The summary is merely highlights, ideas for any whole we can poke in the paper, room for improvements, take-aways, etc. Definitions are at the end.)

**( (** For the accompanying paper, for me, I use a few colors for reasons. Eg., PURPLE means that it highlights new information, and may be a Keyword. PINK highlights note things to be specially noted, or even something that may lead to OUR proposal. GREEN is an emphasized YELLOW, where yellow is an important passage. BLUE is just to highlight yellow highlighted stuff))

**( ( (** Sticky notes spread along the pdf are annotations or even questions I have.)))

ABSTRACT

… phishing toolkits have been *helping* attackers…

[sic] toolkits act as malicious *reverse proxy server*s of online services, mirroring live content to users while extracting credentials and session cookies in transit. End sic.

One operation in the TK, (toolkit) is that they harvest 2FA’s, for netwk level properties.

With 99.9% accuracy, they’ve identified the properties.

[they use] a longitudinal study of **MITM** phishing toolkits by creating a data-collection framework that monitors and **crawls** suspicious **URL**s from public sources.

??? [sic] They discover that MITM phishing toolkits occupy a blind spot in **phishing blocklists**, with only 43.7% of domains and 18.9% of IP addresses associated with MITM phishing toolkits present on blocklists, leaving unsuspecting users vulnerable to these attacks. ???

INTRODUCTION

Vastly so, usernames and passwords are the default gateways the online services have users interact with. Social engineering is used.

[Phishers] make sites [clone-ish], and operable and updatable if the authentic site updates theirs.

As of late, there are all-in-one-phishing-toolkits.

They automatically fetch static copies of webpages, and the toolkits have **cloaking** realtime (for today’s [online experiences]) mechanisms.

While requests and responses are being forwarded, from the malicious proxy server to V, in the meantime stealing creds and cookies.

A **V or v**, (victim, in this paper can be either the client or legal server, a la target.com …)

[sic][The] phishing page is now a “perfect” copy of the victim website) as well as manually communicating with the target website to send the user credentials and 2FA codes to obtain the authenticated session cookie. End sic.

Students study 13 **PTK** (phishing toolkit) versions. ..from perspectives of target site and interacting w the **P (or p)** (phising site.)

Creds are stolen as a result. next-**gen p tk**’s, (toolkits) act as REVERSE PROXY SERVERS to steal creds and session info, (cookies) in transit. The connection is HTTPS.

[In Fig-1, think Duo]

Diagram

Description automatically generated



[sic]We produce a globally-diverse dataset of laboratory MITM phishing toolkit deployments, detailing their network-level characteristics . End sic.

[sic]Using this dataset, we develop a machine learning classifier *that leverages the network timing discrepancies* *inherent to reverse proxy servers* to detect the presence of MITM phishing toolkits with **99.9%** accuracy

PHISHING **BLOCKLIST** services\* web infrastructure can use and leverage PHOCA.

**PHOCA as a fingerprinting MITM tool detector** for analysis of phishing toolkits.

*We want to expand their coverage on MITM toolkits including popular sites to DETECT these attacks of phishing, (malicious REQUESTS) from MITM PHISHING TOOLKITS.*

With the help of Palo Alto Networks, (**PAN)** it was fond that over 6 months, and ~20% of known **p** (phishing) sites received > 6k customer requests. ((Being scammed.))

*Contributions of the paper* :

• Original in-depth research.

• ntwk level features for the ML classifier are derived to classify phishing sites HOSTED by the

toolkits. (99.9% accuracy.)

• PHOCA is developed as a MITM p toolkit fingerprinting framework.

It *classifies* MITM p toolkits and *collects data* on them.

• **Also with PHOCA, they found that current p BLOCKLISTS do *NOT effectively(?) REPORT the***

***malicious sites.***

•s They show that the toolkits can be IDENTIFIED from both the perspectives of

**t and v**, (target (web server )and victim ends).

BACKGROUND

Re; **RPS**’s (REVERSE PROXY SERVERS, see def blw) Usecases are load balancers, [sic] to providing authentication for services in pvt ntwks. They act as MIDDLEMEN (not a bad thing). End sic. They **broker connections** b/t USERS and BACKEND webservers.

MITM PTK’s as RPS’s can target > 1 webservers. They act as [normal] web servers

[sic]These toolkits act as web servers when communicating with victims, and clients when communicating with target web servers. End sic.

The PTK’s PERSIST v’s browser after authentication, ie the user continues browsing as ‘normal’ but THROUGH the **P** SERVER.

One powerful use of these toolkits is to compromise user ac- counts that are protected by 2FA mechanisms .

The **T** server sends a copy of the 2FA code, ((think **Duo** eg)) through a pre-esatablished separate channel eg ans SMS or email or a mobile app/HW TOKEN , and this token generates a code

at a client side.

The client unwittingly submits this code to the PTK, where AGAIN it’s forwarded to the TARGET webserver.

Table 1, (also see §6.1) shows that the NETWORK LAYER fingerprinting is more of an effective layer than the APPLICATION LAYER fingerprinting to have the PTK’s CLOAKING TECHNIQUEs (5 of them) not used.

[sic]When authentication is completed, the session cookie provided by the target web server is saved by the MITM phishing toolkit, enabling attackers to now send authenticated requests in the name of the victim. End sic.

The [attack] is deployed on a webserver, then a link is sent to potential **V**’s.

[sic] Even though in theory, an HTTPS connection between the user and the **phishing server** is *NOT* necessary, modern web browsers show a barrage of warnings for websites that are visited over HTTP, particularly when the user is providing input in **HTML forms**. [A modern browser will show many warnings that the site is not HTTPS] End sic. …

…therefore in this study in this paper, since the user is usually using HTML forms IN HTTP (AOT HTTPS), all the MITM PTK’s that they study use valid TLS certificates for their p pages.

**ALL** requests to the p server, inc MALFORMED ones, and ones towards at nonexistent RESOURCES are forwarded to the target webserver.

[sic] Connections between the phishing server and target web server are made over an **additional HTTPS connection,** where the phishing server takes the role of a web client. End sic.

Attackers have compete ctrl over data in the **application layer**. So the researchers can view the p content. But since A HAS complete ctrl, CONTENT-BASED p detection can be prone to failure.

They ensure that all methods for detection do NOT DEPEND ON the integrity of the PROXIED DATA.

**Network-layer fingerprinting**, on the other hand, analyzes features of the network connection and web server in question, making it effective against *all categories except IP*- based cloaking. **However**, it should be noted that IP-based cloaking ***would bypass*** *any form of detection/fingerprinting* *that originates from an IP address considered to be suspicious from the point of view of an attacker.*

(§3)

[We’ll] describe **THREE** MITM PTK’s. \*\* (see § **3.2** )

TRAINING and VALIDATION processes are detailed used to create a classifier for tk detection.

For the scope of this paper, we define a MITM phishing toolkit as a reverse proxy server that mirrors a target web page to a victim while harvesting credentials, 2FA codes, and web page content in transit.

The important distinction between these toolkits and other phishing tools is the continuous proxying of user traffic to and from the target web server, pre- and post-authentication .

PTK’s share common goals but differ in their FEATURE SETS, which they compete on.They also share common architectures.

**§3.2 (is a discussion on\*\*** 1=**Evilginx** followed by2=**Modlishka** followed by3=**Muraena)**

(Feature-sets differ)

**1/3)**

**Easiest to operate in the CLI to config the p server.**

Hosts its own DNS srvr and creates TLS certs for the Let’s Encrypt API.

It’s only manual configurable.

https://evil.com/xICcxSqs = a tokenized **URL** that m/b present in the REQ to make the p site viewable.

[sic]This significantly lowers the barrier of entry for attackers, allowing even the least

technically adept to launch their own phishing campaigns. End sic.

**It’s the only (evil) tool that can host multiple phising pages at the same time.**

Each individual p webpage is a SUBDOMAIN of the attacker’s primary domain.

A’s attack [sic] actions are cloaked exc, V. This is done by LURES, ie, **URL** tokens.

Lures - §3.2 An example of a **tokenized** **URL** generated by Evilginx is https://evil.com**/xICcxSqs**

The **xICcxSqs** is a **random token**.

[sic]All requests missing valid ((legal))tokens are redirected to a web page of the attacker’s choice.

Tokenized **URL’s** responding with p content can be prevented by **A,** disabling INDIVIDual tokens.

**This is a cloaking mechanism of EVILGINX**.

**BLOCKLISTING** is a [method] that halts the attacks once A is detected.

The feature-set of 1/3) is popular, and antivirus SW providers/products (8 of them) etc, are aware.

2**/3)** Modlishka

This is a **barebones** malicious PTK as MITM. It targets 1 domain at a time, AOT the power of

1/3).

There is no CLI for its config setup.

Attacker must provide a cert. It can only auto-generate its own self-signed certs.

It **can** be configured to remove all security and encryption headers from REQ’s to TARGET webservers.

**3/3)** Muraena

[sic]While **Evilginx** and **Modlishka** require the attacker to manually create configuration files detailing the domains and HTML attributes that should be replaced or removed from requests and responses, **Muraena** automates this process using a web crawler.

It also can automate, once the victim is compromised, the **CHANGING OF PASSWORDS** [to its advantage, and more]. $

$ Necrobrowser, its companion tool, does this by taking **Muraena session cookies** then launching an [sic] instrumented CHROME instance using the CHROME DEVTOOLS PROTOCOL.

This is offers an effective p campaign.

§3.3

[sic: Indentations intentional for keywords]

The unique architecture of MITM phishing toolkits allows attackers to create impersonating web pages that effectively fool victims into providing their credentials.

However, this architecture also introduces discrepancies in packet round-trip times

(RTTs), enabling the fingerprinting of these toolkits at the network level.

As two distinct HTTPS sessions must be maintained to broker communication between the victim user and target web server, the ratio of various packet RTTs, such as a

•TCP (handshakes)

• SYN/ACK requests and

• HTTP GET requests, will be much higher when communicating with a reverse proxy server

than with an origin web server directly. End sic.

(The list directly above is tested for testing the efficacy of the timing discrepancies of the PTKs)

(**Valid** and **Malformed** requests are made to entice a direct response from the toolkit rather than a proxied response from the target webserver – see Fig 2)

The ratio of intercepted attacked malicious traffic to normal tfc is furthur increased WHEN the RPS, (reverse proxy server) furthur increases the RTT.

(**REVIEW Fig-2**) . . .

The group develop an ML based classifier on real-world data, and, each of the MPTK

§3.4 MITM PHISHING TOOLKIT CLASSIFIER

Recall [sic]all content viewed on the client de- vice is at the complete control of the attacker

End sic. Therefore it’s easy to obtain features at the APPLICATION lyr && ADAPTABLE to any evolving changes in the tools inc future tools.

They divide their feature set into :

•Network Timing Features, and

•TLS Library Features.

for Valid and Malformed HTTP requests.

((Recall there are TLS handshakes and TLS certificates))

Self note;

(HTTP) **400 Bad Request** response status code indicates that the server cannot or will not process the request due to something that is perceived to be a client error (for example, **malformed** request syntax, invalid request message framing, or deceptive request routing).

https://developer.mozilla.org/en-US/docs/web/http/status/400 //End self note.

TLS LIBRARY FEATURES §3.4

They use TLS implementations as a distinguishing factor in the research since MPTKs usually don’t use the same web, or RPS S/W as benign websites, and they use DIFFERENT TLS LIBS.

[sic]

They treat each TLS version supported by the current web server as a binary feature in their classifier. Also,

they use the **TLS fingerprinting tool TLS Prober** to identify the TLS library utilized by the current web server based on the format of TLS packets it transmits. [See def below].

**TLS Prober (a TLS fingerprinting tool)** returns a map of TLS libraries to the probability each library is used by the web server in question.

Of the 199 features our classifier is composed of,

• 14 are network timing features, and

• 185 are TLS library features. [See Table 7 in appendix]

Re DATA COLLECTION

Websites are route to either to a web svr directly, or thru a reverse proxy srvr in the form of

CDN’s or or load balancers. ((or both?))

Fig-3 Shows the architecture (physical connections fr Stonybrook to srvrs (or computers) of East coast, Asia, Australia, and a country in Europe. **The framewk (netwk) is for collecting data of MITM PTK’s** . Fig-3 goes w/ Fig-7.

**To effectively distinguish b/t MITM attacks and benign traffic**, GROUND\_TRUTH data was collected from

1•) Non-proxied webpages

2•) RP, (reverse proxy) webpages

3•) MITM PTK webpages

1\*) To get a list of sites that **don’t go thru an intervening proxy** svr, but served directly by a webserver, they use a **heuristc of domains** that points to IP addresses of the CHP’s (cloud hosting providers);

> Digital Ocean\*

> Linode\*, inc small businesses. \*similar to AWS or GCP. They’re hosted by ORIGIN SERVERS.

2•) They (researchers) use public IP addrs from CLOUDFLARE. They use Cloudflare to generate a list of benign sites hosted behind a RPS. They use Reverse DNS Lookups for each IP addr in Cloudflare to acquire a list of Cloudflare managed domains.

3•) Since RTT’s vary across different regions, THEY (the researchers) 30 globally distributed nodes **hosted on AWS**. Each NODE HOSTS ;

> a web CLIENT

> 3 PTKs and

> and Apache webserver – all at the same time. See Fig 3.

Figure 3, allows them to record network timings for many of the potential geographic distributions of victim→ phishing toolkit→ target web server permutations.

[sic]For example, our infrastructure allowed us to obtain measurements that included the three parties (victim, phishing toolkit, and target web server) all being located in North America, as well as the victims being located in Asia, having their traffic proxied by an EU-residing phishing server, onto a US-based target web server.

**In each permutation**, **Node A** sends an HTTP GET request to the port of a phishing toolkit on **Node B**, where it is then forwarded to the Apache web server hosted on **Node C**. Since it is unlikely for a phishing toolkit to exist on the same host as a target web server or victim user, we exclude these scenarios. **This leaves us with n(n−1)(n−2), or 24,360 permutations per toolkit**. Since we record data on three toolkits, we obtain a total of 73,080 network measurements for the evaluated MITM phishing toolkits. End sic.

Re; Fig-1 above, since there are 3 (¥)nodes, there are 24,360 permutations. That number is taken from (¥) n(n-1)(n-2) .

Another consideration is that this experiment was (recall) using 3 tk’s.

So now the num of **NETWORK MESUREMENTS is 73, 080** for evaluating the MPTK’s.

They compile a **ground-truth DS of all facets of;**

• each toolkit’s REQuests and responses, includong

\* ((requests and responses)) benign websites.

That leaves a the 73,080 ntwk requests from the

> MITM toolkit, and

> benign categories for a

**TOTAL of 146,160 data points**.

§3.5 MODEL TRAINING AND VALIDATION

Training and testing datasets were constructed using their compiled ground-truth data. The Phishing : Benign ratio was

1:1 .

The RF classifier was trained with [sic] a minimum sample split of **2 and 100 ESTIMATORS**.

[sic]As a result, we argue that even if attackers are awareº of our tool’s presence, it will not be trivial to evade detection by selectively patching individual features. End sic.

The point to all the text above the above sentence: [sic] is that it takes an ENSEMBLE of features to get to the very high remaining (after removing TLS and timing) accuracy or **97%** .

[Next section § is Mmodel Generalizability – pg 6 of 15] /

º They use NETWORK level features so that it increases the robustness of the classifer.

**ATTACKERS CAN MAKE MODIFICATIONS OF CLASSIFIERS to ward-off FINGERPRINTING ATTEMPTS.**. So they seek to create a classifier that is also **GENERALIZABLE to updates** to existing tk’s as well as ones in the future.This is in addition to of course detecting the three identified tk’s.

Again, they tested their classifier with older SW release versions of the toolkits.

Collectively, **3 MPTK versions were tested** – and by or from;

**• Evilginx** (the easiest to operate)

**• Muraena, and**

**• Modlishka** [see pg 4 in §3.2 above]

The last paragraph before §3.6 explains how they mreasured their classifier’s performance on previously UNKnown MPTK’s. //

**§3.6**

**PHOCA** – M-PTK Detection **¶¶** – Latin for “seal”. A seal uses vibration detection to hunt, by way of a prey’s breathing, hidden creatures for food. Their tool, (PHOCA) hunts MPTK’s using inherent features.

**§4 DISCOVERING**

MITM PHISHING SITES IN THE WILD

[sic] Using **PHOCA, we conduct a large-scale search for MITM phishing toolkits in the wild**. We seek to determine the online presence of these tools, and uncover patterns in their usage. This allows them to **expose the source of phishing campaigns** leveraging these tools, as well as ((know)) targeted users and trademarks, ((facebook.com, someUniversity/someWebpage.edu etc)). End sic.

**¶¶** They designed and implemented a **URL** crawling infrastructure. It visits a lot of potential-to-phishing websites. Recording information ((metadata)) on them is done daily. **Classifications**

**• MPTK or**

**• not an MPTK is done.**

§4.1 Collection

They crawl **URL**S’s from **open [anti]phishing databases;**

• Phishtank

• Openphish

• FB Certificate Transparency Phishing.

They also supplement the above • with their **own Certificate Transparency log parsers** which uses regular expressions to search for **combosquatting domains**.

They ended up using, via Certificate Transparency, 22 trademark domains.

Their CT, (certificate transparency) log parser, as well as FB’s parser, can’t classify MPTK’s, but do search for domain names that seem to impersonate known trademarks, (eg.,youtube, dropox, facebook itself, amazon, paypal, nytimes, google . . . see [sic]appendix ??)

They use all **URL**s and domains from each source as i/p to their crawling infrastructure to locate MPTK sites in the wild.

Reliance on CT and p lists evaluating MPTK’s by analysts and researchers exclude test deployements that may exist. .

================================================🡺🡺🡺🡺🡺🡺🡺🡺🡺🡺🡺🡺

**ZMAP**

🡺🡺🡺🡺🡺🡺🡺🡺🡺🡺🡺

[sic] They argue that setting up a subdomain **or purchasing a domain name that matches the target site and deploying a phishing toolkit there**, **crosses the line between benign and malicious.**

[**FOR FURTHER POSSIBLE RESEARCH FOR US………………………………………………………………………..**

Had we used Internet-wide scanning tools (such as **ZMAP** [35]) to identify MITM phishing toolkits, **we would not be able to reliably differentiate between test deployments and deploy- ments by attackers**. 🡨-------------------- pg 7/15.

End sic.

CT logs apparently limits useage to domain names AOT the full **URL**, (that are usually available in PTK’s). However since PHOCA is network-level based to find MPTK’s, they DON’T require access to p content. So only fingerprinting the p webserver is good enough, not the content.

(( protocol://hostname/filename )) == full. Port# optional.

**Crawling Infrastructure**; [see fig-5]

0/4) Fig5 is a high-level view of their URL crawling infrastructure.

(1/4) Their queue-based system takes as input, URLs from a number of **sources**, including phishing **blocklists** (see pg 2) and also impersonating-domains ((youtu.be.com etc)) found on **the Certificate Transparency logs**.

(2/4) The Figure in this section shows dispatches of one of their crawlers to collect data on the corresponding website in real time.

Each **crawler** consists of two modules:

• • a headless **Selenium browser**, and



• • **PHOCA** worker.

**For each website encountered**, they record the following information:

i) HTML and screenshot of the landing page,

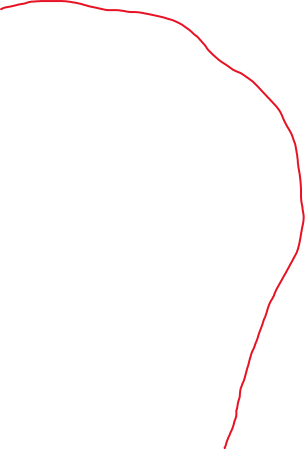
ii) TLS certificate offered to our browser,

iii) original and redirected (if applicable) *domain* ***IP*** *address*, and

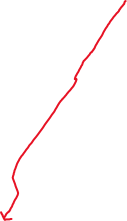
iv) original and redirected (if applicable) ***classification***.

(3/4) This data is **forwarded to** the analysis module which **clusters** web pages based on their content, assisting in the **manual verification** **of the classifications** made by our model. Furthermore, using information such as TLS certificates, web server IP addresses, and domain names allows us to uncover connections between seemingly **independent** phishing websites and **Cluster** individual phishing Sites, **into phishing campaigns**.

(4/4) Our system also utilizes time-based **re-queuing** to record data on websites previously crawled in order to **map the life cycle of MITM phishing websites over time**. We make use of this when crawling domains from the **Certificate Transparency logs**. Since these domains are captured as soon as their certificates are created, there’s a high probability that there will be no web server responding to requests at that moment, so, they re-crawl all such domains periodically **AFTER** their initial recording.



Moreover, they **re-crawl all** **URLs** classified as one of the **pt**’s to measure how long these websites remain active after creation.



they manually sifted thru data of each website labeled as MPTK and removed FALSE positives at the **end** of phase-2.

Using Openphish & Phishtank & **esp FB’s CT-API**, >200 MITM p sites from URLS were reported.



Diagram

Description automatically generated



§4.2 (2 ½ pages)

(Phase-1 is easy to grasp. Page-8.)

(Phase-2 follows. ..and also easy.)

For brevity, they will refer to these two sources as Certificate Transparency for the remainder of this paper. -> They’re referring to ;

• FB CT, and

• Their own (Stonybrook’s) CT.

((**PHOCA** of course is their developed p detector, and not a CT))

Phase-2 completion.

Upon completion of the second phase, they manually inspected the data collected on each website labeled as a MITM phishing toolkit to remove false positives. They did this by analyzing

• **SCREENSHOT**,

• **HTML**, and

• classification data (of each website labeled POS as a p website and confirming the target site is a popular tradmark, (a fake Bank of America, etc), and the NETWORK-LEVEL data matches theprofile of a MPTK. )

Finally identified **5,861 false positives from a total of 7,081 positive classifications**, and promptly removed them from their dataset before conducting any further data analysis.

[sic] Throughout their recording period, **we classified 841,711 web pages**, meaning PHOCA had a 0.6% false positive rate during our data collection period—approximately sixteen per day, on average. End sic.

[**FOR FURTHER POSSIBLE RESEARCH FOR US**:] [sic]

By analyzing the network timings and TLS fingerprints of sedo.com web pages, they infer that this service utilizes a reverse proxy infrastructure with similar net-work timing properties to MITM phishing toolkits. In practice, a web page classification system such as theirs would benefit greatly from a **pre-filtering step** to remove common sources of misclassification.

**MITM PHISHING WEBSITE LIFE-CYCLE** . Still §4.2

The group found that the “sniffing” tool and utility-type website WHOIS, was helpful in their utilization and discovery of how LONG the time was that a MITM p-website were active(?), to when is was uncivered by their tool, (the groups crawlers).

Chart, histogram

Description automatically generated

TARGETED BRANDS AND PHISHING CAMPAIGNS //

PHISHING BLOCKLISTS PRESENCE [sic]

**By querying popular phishing blocklists, we discover that most MITM phishing websites our crawler discovered are missing from these lists. In total, only 43.7% of positively labeled URLs in our dataset are listed as malicious by at least one domain blocklist re- ported by VirusTotal**

End sic.

MITM PHISHING TOOLKIT **CLOAKING**

[sic] Our results demonstrate that due to cloaking and their highly- targeted nature, the phishing websites supported by MITM phishing toolkits **are able to remain hidden** from the scanners that feed popu- lar blocklists. This affords attackers with more time **to inflict damage on a greater number of victims** and decreased resource costs. End sic.

Their results show that p-websites that had high visibility for scrutiny bc of their **cloaking,** made it so the group could not use TOKENIZED URLs.

**So the sites that *were* supported by MPTK’s REMAINED HIDDEN**.

(Recall - PHOCA, being a ntwk crawler and fingerprint tool doesn’t need to see the p-content.).

Ie., they observed a lot of **CLOAKING** **RESPONSES to requests to MITM p-websites in the wild.**

§4.3 CASE STUDY: M P-ATTACKS IN AN ENTERPRISE SETTING.

**THEY PARTNERED WITH PAN, PALO ALTO NETWORKS. They have a large customer base and provide in-line VANTAGE POINTS, and network communications make it so the group can compare their results to.**

MPTKs can not only cloak, but REDIRECT traffic to legit sites.

PAN found that by using **PHOCA**, they (Stony’) identified a few more maliciious p host names.

However, our methodology mitigates these issues by focusing entirely on **ntwk-lvl**

**>>> features.** //

**§5**

SERVER SIDE MITM PHISHING TOOLKIT FINGERPRINTING

**FINGERPRINTING, to ID the type of toolkits used in the wild, are good. The problem IS, is that it’s only good IF AND WHEN the attack is logged into a/the BLOCKLIST**. .



Since data in the application layer is under complete control of the attacker, classic browser-fingerprinting methods will not suffice to determine the presence of a MITM phishing toolkit from the perspective of a targeted web server. For instance, as all JavaScript is executed on the victim’s device, fingerprinting scripts would return information about the victim rather than the toolkit.

We therefore seek to fingerprint MITM phishing toolkits at the TLS layer of the **network** >>> >>> **stack** .

§5.1 TLS FINGERPRINTING

[sic ]However, we are only interested in finding collisions between the JA3 TLS fingerprints of MITM phishing toolkits and web browsers utilized by real users . End sic .

DISCUSSION §6 //

KEY TAKE-AWAYS § 6.1

[sic Classification in the wild: Over the course of our longitudinal study, we discovered 1,220 MITM phishing toolkits targeting pop- ular trademarks such as Google, Facebook, and Yahoo. Moreover, by collaborating with Palo Alto Networks, we identified that enterprise users are being targeted by MITM phishing toolkits .

**Blindspot in phishing blocklists**: The cloaking mechanisms utilized by MITM phishing toolkits severely decrease the effec- tiveness of crowd-sourced blocklists (56.3% of the discovered URLs were missing from all evaluated blocklists).

Phishing block- list services must take a more proactive approach in discovering phishing content. We show that monitoring Certificate Trans-parency logs for impersonating domain names is a successful approach to uncovering otherwise hidden phishing websites.

However, as attackers have full control over application content, payload integrity cannot be ensured. If an attackers knows a particular service uses such application- layer integrity checks, they could simply remove this code prior to sending it to the victim. End sic.

Using U2F (universal) A key is generated and bound to the domain of the online service. Keys generated with a MITM, are invalid.

§ 6.2 LIMITATIONS

[ 1st – for the following LIMITATIONS, recall ] >>>

From **3.4 MITM Phishing Toolkit Classifier** [sic]

Motivated by the results from our exploratory analysis, we develop a machine-learning-based classifier trained on data gathered from real-world websites and each of the MITM phishing toolkits in a laboratory setting.

**Feature Engineering**

As mentioned in Section 2.3, all content viewed on the client device is at the complete control of the attacker, making any features present in the application layer easily modifiable.

Thus, when designing our classifier, we focus on features inherent to the nature of the MPTKs. Using these types of features provides us with a robust and powerful classifier that is not only effective at the time of writing, but is also adaptable to changes in existing tools as well as future tools. ALSO REFER TO TABLE-1 RE: NETWORK LYR vs APPLICATION LAYER fingerprinting technique being = SUCCESS or FAIL(or not as efficient)

To this end, we divide our feature set into **Network Timing Features**, and **TLS Library Features**.

we show in Section 3.4 that training data can be quickly and easily generated to update our classifier to match such modifications. Moreover, as our classifier includes network timing features that are consistently present in reverse proxy-server deployments, it is agnostic to many modifications made by attackers.

Lastly, while we show that our fingerprinting technique is highly effective against MITM phishing toolkits, it is unable to discover traditional phishing websites. This is a strength of visual phish- ing detection compared to our approach. However, we note that PHOCA can be easily implemented into existing anti-phishing ser- vices, and should be used as an additional tool beside visual phishing detection, rather than a replacement to it.

RELATED WORK //

CONCLUSION

CITATIONS //

**KEYWORD**s **///////////**

**/////////////////////////////////////////////////////////////////////////////////////**

**/////////////////////////////////////////////////////////////////////////////////////**

**MPTK**s = man-in-the-middle phishing toolkits.

**Tool** – Apparently, this can refer to either the phishing tool, or THEIR tool, PHOCA. Caution!

**PHOCA** (their “tool”) – see §3.6

**Gatekeeper** - eg usr and passwd. \*

**Social-engineering** – eg to acquire the prized, \*. The creds were then stored in the **A**’s DB

or revealed to the **V** that they now know their \*.

**Phishers** (**p’s**) – make sites clone-ish, and operable and updatable if the authentic site updates

theirs.

**//////////////////////**

**MPTK**’s:

• Evelginx (inc its tokenized **URL**s)

• Modlishka,

• Muraena

**RPS’**s– reverse proxy servers - **front-end web servers that perform all DIRECT communication with each web client.** They are utilized in a variety of use cases from **load balancing**, to providing **authentication** for services residing on private networks. These servers act as “middlemen,” brokering connections between users and backend web servers.

**//////////////////////**

Typically, **TLS** connections are terminated at reverse proxy servers, thereby decreasing the complexity of configuring TLS certificates for website administrators by creating a single con- figuration point. Some of the most popular RPS’s are: *Squid, Nginx, and Apache Traffic Server*.

**Chrome Devtools protocol**

***sic*** – sic errat scriptum – as it were. Eg., He said “…and Rebel said, “yo!””.

**Ld balancers**.

**TLS** – certs and TLS handshakes

**Token**, WRT §2.2

**Cloaking**. See §6.1 as well as Table-1.

**Link** – ITO the attacker’s link when the attack from the PTK is deployed on unsuspecting

webservers that now contain the attack.

(( $ ping -c 5 video.pbs.org ))

Resource – see §2.3

**Cleartext** – see §2.3 still. transmitted or stored text that has not been subjected to encryption and is not meant to be encrypted

.

**//////////////////////**

**INLINE ACCESS TO NETWORK COMMUNICATION** – see §2.3

**Content-based** – 2.3

**Proxied data** – see 2.3

**Network-layer** and **application-layer** fingerprinting – 2.3

except IP- based cloaking – 2.3

**Clear-web**– AKA Clearnet or Surface Web. This is **AOT Dark web or Deep web**.

**PTK’**s share common goals but differ in their FEATURE SETS, which they compete on.

**Cloak(ing)**

**Lures** - §3.2 An example of a tokenized **URL** generated by Evilginx is https://evil.com/xICcxSqs,

**Subdomain** – …

**BLOCKlisting** – **HALTING of eg.,, Evilginx’s attacks.**

**Web crawler** - §3.2 – eg to automate a config file to be created. Also,

sometimes called a spider or **spiderbot** and often shortened to crawler, is an Internet bot that **systematically browses the Web** and that is typically operated by search engines for the purpose of **web indexing** .

**//////////////////////**

**RTT**’s and Fingerprinting- §3.3 – round trip times

**RPS** (reverse proxy server)

**TLS** Client Hello packets

**TLS Prober** – a TSL fingerprint tool. It determines the lob used by a webserver via a series

of TLS-CLIENT-Hello packets.

**Cipher Suites**

**TLS stack**

**Toolkit-agnostic**

**Load balancers**

**CDN**s

**p** - phishing

**tk** – toolkit

**CHP**’s – cloud-hosting providers. GCP, Azure, AWS, Oracle …

**//////////////////////**

**Ground-truth dataset** - https://medium.com/wao-ai/dataset-vs-ground-truth-dataset-94c565b0d4b6 It’s a regular DS but with annotations. The data samples can contain many different things to **help the ML algorithm**. A regular dataset has nothing inside the samples.

A screenshot of a computer

Description automatically generated with low confidence

**Heuristic of domain**

**CHP**’s – cld hosting providers

Cloudflare – one of the most popular anti-DDoS CDN services.

**Reverse DNS lookups**

**THEY (the researchers)**

**Permutation**

**Estimators** (**or n\_estimators**) – medium.com – [sic]It takes an integer value which represents the number of decision trees the algorithm builds. In general, a higher number of trees increases the performance and makes the predictions more stable, but it also slows down the computation. The default value of this parameter is 100.

**//////////////////////**

**Hyperparameters** – see below Table-2. (This may allude to to Fig-4, where these HYPERparameters MAY refer to

• TLS Features and

• Ntwk timing features.

**Generalizable** (ITO a model) – eg used in pg 6/15

**URL crawling** infrastructure – Noted in several places. In §4.1, it says that this visits p websites as they’re created . It crawls and collects [metadata] about each site

**and a label using PHOCA** ??

• **Phishtank** – a new type of OpenDNS – see §4.1 ((a l thinktank))

• **OpenPhish** – [sic]is a fully automated self-contained **platform** for **phishing intelligence.** **It identifies phishing sites and performs intelligence analysis** in real time without human intervention and without using any external resources, such as blacklists. **OpenPhish** receives millions of unfiltered **URL**s from a variety of sources on its global partner. End sic.

• Also, **Certificate Transparency** logs (a log parser) – a la Facebook (Meta) Cert’n Transparency **Phishing API**.

**Regular Expressions** – see §4.1

**//////////////////////**

**Combosquatting domains** – [sic]Domain squatting is a common adversarial practice where attackers register domain names that are purposefully similar to popular domains. **In this work, we study a specific type of domain squatting called "combosquatting,"** in which attackers register domains that combine a popular trademark with one or more phrases (e.g., **betterfacebook.com, youtu.be , etc.**

**TYPOsquatting** aka URL hijacking, a sting site, or a fake URL, is a form of cybersquatting, and possibly brandjacking which **relies on mistakes such as typos** made by Internet users when inputting a website address into a web browser.

**Zmap** (sp correct) -  a fast single-packet **network scanner** optimized for Internet-wide network surveys. On a computer with a gigabit connection, **ZMap** can scan the entire public IPv4 address space in under 45 minutes. With a 10gigE connection and PF\_RING, **ZMap** can scan the IPv4 address space in 5 minutes. [https://zmap.io](https://zmap.io/)

**TLS certificate** – Transport Layer Security (TLS) certificates—most commonly known as SSL, or *digital certificates*—are the foundation of a safe and secure internet. **TLS**/(ssl) certs secure internet connections by encrypting data sent between your browser, the website/website server. They ensure that data is transmitted privately and without modifications, loss or theft.

TLS fingerprints - “…JA3 TLS Fingerprint …” mentioned in §5.1.

**FALSE positives**.

Domain parking services -  Even if you are not ready to build and launch your website, if the domain name you’ve chosen is available, it is in your best interest to register it now. This practice is called domain parking. GoDaddy, sedo.com and HostPapa offer it.

**//////////////////////**

**In practice, a web page classification system such as ours would benefit greatly from a pre-filtering step to remove common sources of misclassification.**

**AUTONOMOUS SYSTEM** [sic – certainly.io]

Traditionally, e-commerce businesses have used cookies and purchase history in their efforts to **personalize customer journeys** and experiences.

However, with the introduction of the autonomous web, cookies will become obsolete, and a de facto autonomous database will replace them.

We call it **zero-party data**, which comprises anything a customer intentionally and proactively shares with a brand.

Zero-party data presents an opportunity for brands to run effective campaigns which build on **trust and transparency** while simultaneously collecting data from the source.

**gTLD** - .com, .edu, .org, …

**TLD** - .us, .co, .ca …

**//////////////////////**

**TARGET EMBEDDING** - consider the target embedding domain

“www.facebook.com.**user-29de84ca4bfa72.tk**”. The target, in this case “facebook.com”, is embedded using subdomains of **the actual domain, “user-29de84ca4bfa72.tk**”.

The target’s **TLD** can also appear in the real **e2LD**, (effective 2nd level domain) such as apple.**com**-login.pw.

((((**Target Encoding** – [kaggle.com] any kind of encoding that replaces a feature's categories with some number derived from the target.

A simple and effective version is to apply a group aggregation from Lesson 3, like the mean.Using the *Automobiles* dataset, this computes the average price of each vehicle's make:

autos["make\_encoded"] = autos.groupby("make")["price"].transform("mean")

autos[["make", "price", "make\_encoded"]].head(10)

))))

**HMAC** – Hash-based authorization code

**TOTP** – Time-based on-time password, HMAC based.

**HOTP** – Event-based one0time password, HMAC based.

Graphical user interface, text, application, chat or text message

Description automatically generated

**//////////////////////**

**VirusTtal** – Spanish website. Aggregates many antivirus products and online scan enginescalled Contributors. The aggregated data from these Contributors allows a user to check for viruses that the user's own antivirus software may have missedin-line vantage point .

**PAN** – Palo Alto Ntwks – Stonybrook teamed up w PAN to compare to. PAN employes various scanners. The grp shows that their tool (phoca) adds a SIGNIFICANT IMPROVEMENT to PANs toolset.

REDIRECTION – redirecting. Today, cloaking and this are utilized my attackers’ MPTKs.

**FINGERPRINTING - to ID the type of toolkits used in the wild, used for investigations.**

ADVERTISING IMPRESSIONS :

Graphical user interface, text, application

Description automatically generated

**//////////////////////**

§5.2 SERVER SIDE TLS FINGERPRINTINT RESULTS

**A3 TLS** fingerprints (inc the dataset).

ja3er.com - fingerprint database.

User-Agents – a la browers, email readers . . .

Proxying – see §6.1

Longitudinal study – see §6.1

**U2F** Universal Two Factor - §6.1

**//////////////////// END Key words///////////////////////////////////////////////////**