Opening the Blackbox of VirusTotal: Analyzing Online Phishing Scan Engines

Peng Peng[*], Limin Yang[‡], Linhai Song[†], Gang Wang[‡]
[*]Virginia Tech [†]The Pennsylvania State University [‡]University of Illinois at Urbana-Champaign
pengp17@vt.edu, liminy2@illinois.edu, songlh@ist.psu.edu, gangw@illinois.edu

ABSTRACT

Online scan engines such as VirusTotal are heavily used by researchers to label malicious URLs and files. Unfortunately, it is not well understood how the labels are generated and how reliable the scanning results are. In this paper, we focus on VirusTotal and its 68 third-party vendors to examine their labeling process on phishing URLs. We perform a series of measurements by setting up our own phishing websites (mimicking PayPal and IRS) and submitting the URLs for scanning. By analyzing the incoming network traffic and the dynamic label changes at VirusTotal, we reveal new insights into how VirusTotal works and the quality of their labels. Among other things, we show that vendors have trouble flagging all phishing sites, and even the best vendors missed 30% of our phishing sites. In addition, the scanning results are not immediately updated to VirusTotal after the scanning, and there are inconsistent results between VirusTotal scan and some vendors' own scanners. Our results reveal the need for developing more rigorous methodologies to assess and make use of the labels obtained from VirusTotal.

CCS CONCEPTS

• **Security and privacy** → *Web application security.*

ACM Reference Format:
Peng Peng[*], Limin Yang[‡], Linhai Song[†], Gang Wang[‡]. 2019. Opening the Blackbox of VirusTotal: Analyzing Online Phishing Scan Engines. In *Internet Measurement Conference (IMC '19), October 21–23, 2019, Amsterdam, Netherlands.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3355369.3355585

1 INTRODUCTION

Online scan engines, designed to scan malware files and malicious websites, are critical tools for detecting new threats [3, 4, 7, 8]. VirusTotal is one of the most popular scanning services that are widely used by researchers and industry practitioners [8]. VirusTotal provides both file scan (for malware analysis) and URL scan services (for detecting phishing and malware hosts). It works with more than 60 security vendors to aggregate their scanning results.

IMC '19, October 21–23, 2019, Amsterdam, Netherlands

ACM ISBN 978-1-4503-6948-0/19/10…$15.00
https://doi.org/10.1145/3355369.3355585

VirusTotal is widely used by the research community for data labeling or system evaluation. Many recent works rely on VirusTotal's file scan API [18, 24, 26, 28, 29, 37, 41, 44, 45] and the URL scan API [16, 17, 20, 30, 35, 36, 38, 40, 42, 46, 47] for data labelling. For example, if a certain number of vendors label a file/URL as "malicious", researchers would take the "malicious" label.

Unfortunately, VirusTotal works like a blackbox, and it is not well understood how VirusTotal and its vendors generate the labels for a given URL or file. This leads to critical questions: *are these labels even reliable? Are researchers using VirusTotal in the right way?*

In this paper, we take the initial steps to explore how VirusTotal and its vendors assign labels. We specifically look into how the URL scan API detects *phishing websites*. Focusing on phishing scan allows us to design more focused experiments. At the same time, we seek to design a methodology that can be adapted to other applications (*e.g.*, file scan for malware analysis). We want to explore (1) how VirusTotal works with 68 vendors to perform URL scanning and result updating; (2) how effective these scanners are in detecting simple and more advanced phishing sites; (3) how the scanners react to the dynamic changes of phishing sites. The goal is to provide insights to guide practitioners to better use VirusTotal.

To answer these questions, we set up a series of phishing websites of our own with freshly registered domains. By submitting the phishing URLs to various scan APIs (VirusTotal's APIs and some vendors' own APIs), we collect the incoming network traffic to our phishing sites. At the same time, we continuously query the labels for these URLs from VirusTotal over a month. We set up two types of phishing pages that impersonate PayPal [6] and IRS (Internal Revenue Service) [2] respectively. Across all the experiments, we have used 66 experimental websites in total. We find that:

- *First*, most vendors have trouble detecting the simple phishing sites we set up. Over multiple scans, only 15 vendors (out of 68) have detected at least one of the 36 simple phishing sites. The best vendor only detected 26 simple phishing sites.
- *Second*, the detection performance is drastically different for different phishing sites. The PayPal sites (as a popular target of phishing) can be detected quickly by more than 10 vendors during the first scan. However, the less common IRS sites cannot be detected by any of the 68 vendors using the VirusTotal scan API alone.
- *Third*, the scanning results of vendors are not updated to VirusTotal immediately after the scanning is finished. The delay is caused by the fact that VirusTotal only pulls the previous scanning results when a new scan request is submitted for the same URL. A user who simply calls the query/report API once would not get the updated scanning results.

- *Fourth*, VirusTotal has inconsistent results with the vendors' own scan APIs. The result suggests that third-party vendors do not always give VirusTotal the scan permission or the most updated blacklists.
- *Fifth*, we also tested the effectiveness of obfuscation (methods used by attackers to modify the website to deliberately evade detection). We find that VirusTotal scanners can handle some obfuscation techniques (such as URL shortening), but can be fooled by other obfuscation methods such as image-based or code-based obfuscations.

Our work is an initial step to investigate the reliability of labels obtained from online scanners. Future work can look into other types of scanning (*e.g.*, malware scanning), and measure the correlation between the labels from different vendors, and develop new methods to reliably aggregate different scanning results. To facilitate future research, we release the collected datasets for sharing[1].

2 BACKGROUND & RELATED WORK

VirusTotal APIs. VirusTotal is a popular service that scans malicious files and web URLs [8]. The URL scanning, in particular, aims to detect websites that deliver malware or perform phishing. As shown in Figure 1, VirusTotal works with 68 third-party security vendors (see the full list at [11]). After an URL is submitted to VirusTotal through the *scan API*, VirusTotal pass the URL to these vendors (*i.e.*, anti-virus engines or online scanning services). The scanning results will be stored in the VirusTotal database.

VirusTotal provides another *querying API* (or report API) which allows people to query the VirusTotal database to check if an URL is malicious [10]. Given a URL, the API returns the labels from all the vendors that have previously scanned the URL (and the timestamp of the scanning). It is not uncommon for vendors to disagree with each other. For example, a URL might be labeled as "benign" by Google Safe Browsing, but is labeled as "malicious" by Kaspersky.

Third-party Vendors. Among the 68 third-party vendors, we find that 18 of them also provide their own scan APIs to the public. Table 1 lists the names of the 18 vendors. As shown in Figure 1, these 18 vendors can either be reached via the VirusTotal scan APIs or via their own APIs directly. For a given vendor, it is not quite clear if the two APIs return consistent results.

Using VirusTotal for Labeling. VirusTotal has been heavily used by the research community to label both malicious files [18, 23, 24, 26, 28, 29, 37, 39, 41, 44, 45] and suspicious IPs and URLs [16, 20, 30, 33, 35, 36, 38, 40, 42, 46, 47]. A closer examination shows that VirusTotal is used in different ways by researchers.

First, given that vendors often do not agree with each other (or some vendors have never scanned the URL), researchers need to aggregate the labels to determine if the URL is "malicious". Recall that given a URL, more than 60 labels are returned from the vendors via VirusTotal. We find that most papers define a *threshold* — if at least t vendors return a "malicious" label, then the URL is regarded as malicious. Most papers set $t = 1$ [16, 17, 20, 30, 35, 36, 40, 47], while a few papers are more conservative by setting $t = 2$ or 3 [33, 38, 42] (sometimes researchers use a bigger threshold such as 40 when labeling *malware* files [15, 26]). Second, given a vendor,

[1] https://github.com/whyisyoung/VirusTotal

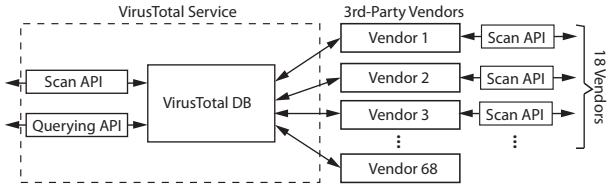


Figure 1: VirusTotal and third-party security vendors.

CyberCrime, Dr.Web, Forcepoint, Fortinet,
Google Safe Browsing, Kaspersky, malwares.com, Netcraft,
NotMining, Phishtank, Quttera, scumware.org, securolytics,
Sucuri Site Check, URLQuery, ZeroCERT, ZeusTracker, zvelo

Table 1: 18 VirusTotal vendors provide their own scan APIs.

its internal model may be updated over time, and thus the labels on URLs may also change over time. Kantchelian et. al investigate this issue for the file scan API [22], and show that one needs to wait for a while before a label gets stabilized. It is unknown if the same issue applies to URL scan.

Phishing Blacklist. Our work is also related to those that study phishing blacklists [12, 32, 43]. Phishing blacklists often have major delays in blocking new phishing sites [14, 19, 31], and suffer from incomplete coverage [13]. Different blacklists may return inconsistent results [25]. Our work aims to look deeper into the process of how phishing URLs get blacklisted (*i.e.*, URL scanning) by VirusTotal and its vendors. The most relevant work to ours is [32]. The differences are two folds: First, [32] looks into the phishing blacklists used by different browsers (*e.g.*, Chrome, Safari), while we focus on how phishing blacklists are aggregated by VirusTotal. Second, [32] focuses on the *cloaking techniques* used by phishing sites, while we focus on the performances of different vendors (scanners), and their consistency.

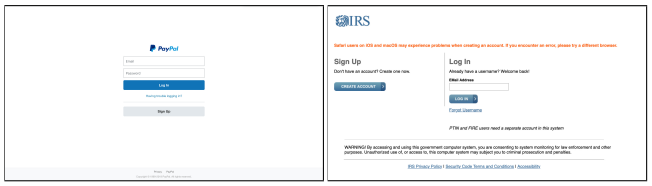
3 METHODOLOGY

In this paper, we want to understand how VirusTotal and its vendors scan phishing URLs. We ask key questions regarding how the labels should be interpreted and used: (1) how effective are VirusTotal's vendors (scanners) in detecting basic phishing pages? (2) how quickly will the scanning results become available? (3) how consistent are the scanning results across vendors, and between vendor-APIs and VirusTotal API? (4) how quickly can VirusTotal react to phishing site changes such as take-down? (5) how much do basic obfuscation techniques help with evading the detection?

To answer the questions, we set up fresh phishing websites on newly registered web domains. Then by submitting the phishing URLs to VirusTotal, we collect incoming network traffic to the phishing servers and the VirusTotal's labeling results for these URLs. We have carefully designed the experiments to ensure research ethics. We have a detailed discussion on ethics in the Appendix.

3.1 Phishing Site Setups

Phishing Page Content. As shown in Figure 2, we create two phishing pages that mimic the login pages of PayPal [6] and IRS (Internal Revenue Service) [2]. PayPal is chosen for its popularity —



(a) PayPal (b) IRS

Figure 2: Screenshots of experiment phishing pages.

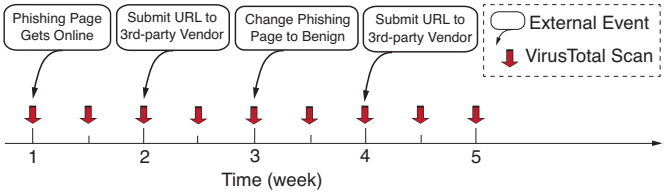


Figure 3: Illustration of the `main` experiment on a given phishing site. The third-party vendor is one of the 18 vendors that provide their own scan APIs.

more than 30% of phishing URLs at major blacklists are targeting PayPal [34]. IRS, as a comparison baseline, is not commonly targeted. We replicate the original sites of PayPal and IRS, and modify the login form so that login information will be sent to our servers. By default, we disable any form of cloaking for the phishing sites. Cloaking means a phishing site hides itself by showing a benign page when it recognizes the incoming request is from a known security firm [21, 32]. The `robots.txt` is also set to allow web crawlers to access the phishing page.

Domain Names. We register *fresh* domain names for our phishing sites. This is to make sure the domain names do not have any past history that may interfere with the measurement. To prevent innocent users from mistyping the domain names (*i.e.*, accidentally visiting our websites), we register long random strings as domain names (50 characters each) from NameSilo [5]. For example, one of the domain names is "`yzdfbltrok9m58cdl0lvjznzwjjcd2ihp5pgb295hfj5u42ff0.xyz`".

Web Hosting. We host the phishing websites at a web hosting service called Digital Ocean [1] on static IPs. Before the experiment, we made sure all the IPs and domain names are publicly accessible, and are not blacklisted by any major blacklist. We have informed Digital Ocean of our research, and have received their consent.

3.2 Experiment Design

The experiments were conducted from March to April in 2019, including a *main* experiment and a *baseline* experiment.

Main Experiment. The main experiment is designed to measure (a) the phishing detection accuracy of VirusTotal and vendors; (b) the potential inconsistency between VirusTotal API and the vendors' APIs; (c) the reaction of VirusTotal to changes of phishing sites. Recall that there are 18 vendors that have their own scan APIs. To accurately capture their impact, we set up separate phishing sites (1 PayPal and 1 IRS) for each vendor (36 sites in total).

For each phishing site, we conduct a 4-week experiment as illustrated in Figure 3. We periodically submit the phishing URL to VirusTotal's *scan API*. The VirusTotal scan API will trigger the scanning of (some of) the third-party vendors. VirusTotal scanning is conducted twice a week on Mondays and Thursdays. At the same time, we schedule 4 external events (on the Mondays of each week):

(1) **Week1**: We put the phishing site online.
(2) **Week2**: We submit the phishing URL to one of the 18 vendors who have their own scan APIs.
(3) **Week3**: We take down the phishing page, and replace it with a benign page (*i.e.*, a blank page).
(4) **Week4**: We submit the phishing URL to the same third-party vendor as week2.

Note that (2) and (4) are designed to measure the consistency between VirusTotal scanning and the vendors' own scanning. Each phishing site is only submitted to one vendor API so that we can measure the differences between vendors.

During the experiment, we collect two types of data. First, we collect the *labels* for all the phishing URLs using VirusTotal's querying API. Note that after a URL is submitted for scanning, the scanning results (*i.e.*, labels) might not be immediately available in the VirusTotal database. So we crawl the labels every 60 minutes to track the fine-grained dynamic changes. Second, we log the incoming network traffic to all of the phishing servers.

Baseline Experiment. The baseline experiment is to measure the long-term reaction of VirusTotal after *a single VirusTotal scan*. We set 2 additional phishing sites (PayPal and IRS) and only submit the URLs to VirusTotal scan API for once at the beginning of the first week. Then we monitor incoming traffic to the phishing servers, and query the VirusTotal labels in the next 4 weeks.

Summary. In total, 38 websites are set up for our experiments (36 for main, 2 for baseline). There are 19 PayPal sites and 19 IRS sites. All the PayPal sites have identical web page content (hosted under different domain names). All the IRS sites share the same content (with different domain names).

Exp.	Brand	# Requests Avg. (STD)	# IPs Avg. (STD)	# Mal. Labels Avg. (STD)
Main	PayPal	12,327 (2,478)	2,384 (290)	16.6 (1.1)
	IRS	335 (364)	146 (107)	13.5 (0.5)
Baseline	PayPal	6,291	2,033	0
	IRS	30	26	0

Table 2: The number of incoming network requests to fetch the phishing URLs, and the unique number of IPs *per phishing site* over the 4-week period. We show the average number of total "malicious" labels from VirusTotal per phishing site (if a vendor once gave a malicious label and then changed it back later, we still count it).

4 MEASUREMENT RESULTS

Our measurement returns a number of important results.

(a) Incoming Network Traffic. Table 2 shows the statistics the incoming network requests that fetch the phishing URLs. Clearly, PayPal sites have received significantly more network traffic than IRS sites. On average, each PayPal site has received more than 12,000 requests while an IRS site has only received 335 requests.

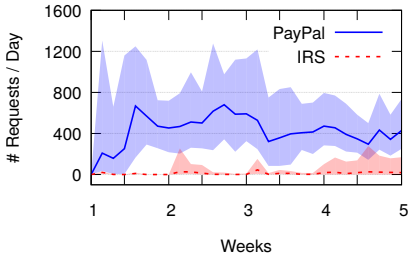


Figure 4: The number of incoming network requests per day per phishing site (main experiment).

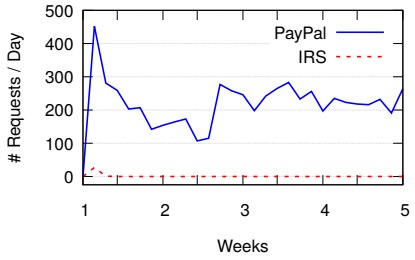


Figure 5: The number of incoming network requests per day per phishing site (baseline experiment).

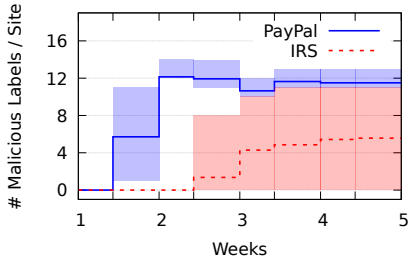


Figure 6: The average, maximum, and minimum number of malicious labels per site (main experiment).

As shown in Figure 4, IRS sites barely have any traffic in the first week, and only start to receive more traffic in the second week.

Interestingly, the traffic volume is correlated with the "labels" received by the sites. Figure 6 shows the number of VirusTotal vendors that flagged a phishing site as malicious (*i.e.*, number of malicious labels per site). PayPal sites get flagged by some vendors right away in the first week, while IRS sites are only detected at a much later time (after vendor API scan). The hypothesis is that after a phishing site is flagged by *some vendors*, then it will be shared with other vendors to perform more in-depth scanning. Figure 5 further confirms this intuition. For the IRS site (baseline experiment), we only submit its URL to VirusTotal scan for once which failed to detect it. Then there is almost no more traffic in the following weeks. The PayPal site, since it got flagged after the scan, continues to receive incoming traffic.

After looking into the traffic log, we notice that not all the requests are pointed towards the submitted phishing URLs. Some scanners also attempted to retrieve the resources under the root directory ("/") or non-existing pages such as "payload.php" or "shell.php". For example, in the baseline experiment, the PayPal site has received 6,291 requests for the phishing URL (see Table 2), and 19,222 requests for other URLs or resources. This indicates that the scanners are looking for signs of malware hosting or website compromise.

(b) Delay of Label Updating. A closer examination of Figure 6 shows that VirusTotal has a delay of updating the labels to its database. More specifically, the x-axis in Figure 6 is the label querying time (label crawling is done every hour). We observe that only after the second VirusTotal scan will the first scan result get updated to VirusTotal database.

For example, in the first week, we submit the PayPal URLs to VirusTotal on day-1. The querying API returns "benign" labels since these URLs were never scanned before by any vendor. Then after we submit the URLs again on day-4, the querying API starts to return "malicious" labels from some vendors. Based on the "scanning time" on the returned labels, we see that theses "malicious" labels are actually originated from the scan of *day-1*. This means, although some vendors have already detected the phishing page on day-1, the results would not be updated to VirusTotal database until the next scan request on day-4.

The result shows VirusTotal uses "pull" (instead of "push") to get scanning results from vendors. The pull action is only triggered by VirusTotal's scan API but not the querying API. Our baseline

Vendor Name	Brand	VTotal Before	Vendor (week-2)	VTotal After
Forcepoint	PayPal	**0**	**1**	**0**
Sucuri Site Check	PayPal	**0**	**1**	**0**
Quttera	PayPal	**0**	**1**	**0**
URLQuery	PayPal	**0**	**1**	**0**
ZeroCERT	PayPal	**0**	**1**	**0**
Fortinet	IRS	**0**	**1**	**0**
Google Safe Brows.	PayPal	**0**	**1**	**0**
	IRS	**0**	**1**	**0**
Netcraft	IRS	**0**	**1**	**1**

Table 3: Inconsistent labels between VirusTotal scan and Vendor scan. "1" means malicious and "0" means benign.

experiment further confirms that vendors do not proactively push new results to VirusTotal database. In the baseline setting, we only submit the URL to VirusTotal on day-1 without any further actions. By querying the labels for the next 4 weeks, we confirm that the scanning results are never updated back to VirusTotal. If a researcher only scans a URL once and queries the database afterward, she cannot get the updated labels. Instead, the researcher needs to perform *two* scans: one for URL scanning, and the other for triggering the database update.

(c) PayPal vs. IRS. The VirusTotal scan during week-1 failed to detect any IRS website (Figure 6). After we directly submitted the IRS URLs to individual vendors, some of the IRS pages started to be flagged. On the contrary, all PayPal sites were flagged by at least 10 vendors during week-1. One hypotehsis is that PayPal phishing pages are more common, and thus the vendors' models (*e.g.*, classifiers) are better trained to detect them. To validate this hypothesis, more rigorous tests are needed by testing a wide range of phishing pages (content), which is part of our future work.

(d) VirusTotal vs. Vendors. The vendors' own scan APIs and VirusTotal's scan APIs do not always return consistent results. Note that when we use the vendor's API, the API returns the scanning result too. Unlike VirusTotal API, vendors' own APIs ask the user to wait until the scanning is finished so that the user gets the real scanning result. In Table 3, we show the vendor API results (on Monday of week 2), and the VirusTotal labels right before and after that (results for the Thursday of week 1 and the Thursday of week 2 respectively). We have considered the delay of label updating of VirusTotal and manually aligned the scan time accordingly.

Rank	Vendor	Total	PayPal	IRS
1	Netcraft	26	14	12
2	Emsisoft	26	14	12
3	Fortinet	26	14	12
4	Sophos	23	14	9
5	CRDF	17	14	3
6	Malwarebytes hpHosts	15	14	1
7	BitDefender	15	14	1
8	ESET	15	14	1
9	G-Data	14	14	0
10	Kaspersky	13	1	12
11	Phishtank	10	10	0
12	CyRadar	8	5	3
13	Avira	6	0	6
14	CLEAN MX	6	4	2
15	Trustwave	3	3	0

Table 4: A list of all the vendors that successfully detected the phishing pages (during the first 2 weeks).

As shown in Table 3, there are in total 8 vendors that show inconsistent results. Most vendors have a "0-1-0" pattern for PayPal sites including `Forcepoint`, `Sucuri`, `Quttera`, `URLQuery`, `ZeroCERT`, and `Google Safe Browsing`. This means through VirusTotal scan, these vendors return the label "benign", even though their own scan APIs can detect that the page as "malicious". A possible explanation is that these vendors did not give VirusTotal the permission to trigger their scanners. Instead, VirusTotal runs stripped-down versions of the scanners [9, 27], which cannot detect the phishing page.

For IRS pages, we show that `Fortinet`, `Google Safe Browsing`, and `Netcraft` have detected these IRS pages via their own scan APIs. However, only `Netcraft` has shared this result to VirusTotal after the scan. It should be noted that we have tried to analyze which scanners indeed visited the phishing sites. This attempt failed because scanners were actively hiding their identity by using proxies and cloud services (see §5). Overall, the result shows the VirusTotal does not always reflect the best detection capability of a vendor. If possible, researchers should cross-check the results with individual vendors' APIs.

(e) Detection Accuracy of Vendors. In Table 4, we list all 15 vendors that detected at least one phishing site during the first two weeks (we took down the phishing pages after week-2). We show that even the best vendors cannot detect all phishing sites. The most effective vendors such as `Netcraft` flagged 14 (out of 18) PayPal pages and 12 (out of 18) IRS pages. It is not clear why some sites are not detected given that all 18 PayPal (IRS) sites have the identical content (except for using a different random string as the domain name). In addition, we observe that some of the vendors always flag *the same subset of phishing sites*. For example, `Netcraft`, `Emsisoft`, and `Fortinet` flagged the same 26 sites. Similarly, `Malwarebytes`, `BitDefender` and `ESET` flagged the same 15 sites. This indicates the possibility that certain vendors would copy (synchronize with) each other's blacklist. To validate this hypothesis, more rigorous experiment is needed in future work.

(f) Reaction to Phishing Take-down. We observe that vendors do not quickly take a URL off the blacklist after the phishing site is taken down. On the Monday of week-3, we took down all the phishing pages and replaced them with benign pages. However,

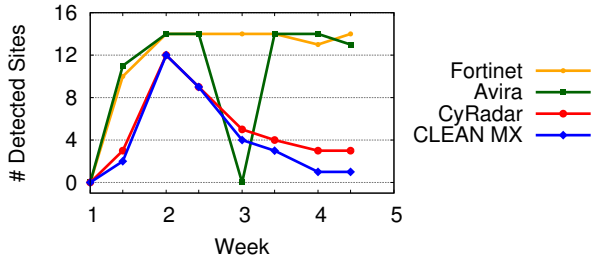


Figure 7: Four vendors have a sign of reaction to the phishing take-down (PayPal sites).

Figure 6 shows the number of malicious labels does not drop even after multiple re-scans.

After examining the results for each vendor, we find 4 vendors that flip some "malicious" labels to "benign" after the third week (for PayPal sites only). Figure 7 shows these 4 vendors and the number of phishing sites they flagged over time. `CyRadar` and `CLEAN MX` already started to flip their malicious labels in week-2 (before phishing take-down), which is not necessarily a reaction to the take-down. `Fortinet` flipped the label on one site in week-4. `Avira` is likely to be reacting to the take-down since it changed all "malicious" labels to "benign" right after the event. Interestingly, the labels were quickly reversed to "malicious" in the next scan.

5 OTHER CONTROLLED EXPERIMENTS

Our experiments lead to new questions: which vendors have indeed visited the phishing sites? What would happen if a phishing site applies simple obfuscation techniques or sets "`robots.txt`" to prevent crawling? How well can VirusTotal detect benign pages? To answer these questions, we conduct additional controlled experiments by setting up 27 new sites.

Vendor Identification. Vendor identification based on the network traffic is very difficult. On average each phishing site was visited by more than 2000 unique IPs (PayPal, Table 2). Leveraging the `whois` records, User-Agents, and the known IP ranges of security vendors, we only successfully confirmed the identity of 5 vendors, including `Dr. Web`, `Forcepoint`, `Google Safe Browsing`, `Quttera`, and `ZeroCERT`. We also tried more controlled experiments by submitting URLs to each of the 18 vendors (one URL per vendor). Even so, we cannot build a reliable identifier for all 18 vendors. The reason is that most vendors route their traffic via proxies or cloud services. The IP set of each vendor dynamically changes too. 32.9% of the traffic comes from known cloud services such as Amazon, Digital Ocean, M247, Feral Hosting, and Linode. It is likely that security vendors are trying to hide their identity to overcome the cloaking of phishing sites [32].

Additional Experiments on Label Updating. So far, our main experiment shows that it takes *two* scan requests to push the scanning results back to VirusTotal (§4(b)). However, the previous experiment is limited to new URLs that are never detected by vendors before. A follow up question is, what if the URL is already blacklisted by the third-party vendor? Do we still need two requests to push the label to VirusTotal? To answer this question, we performed a small controlled experiment. We set up three fresh PayPal

Obfuscation Method	# Sites (PayPal)	Malicious Labels Per Site Min.	Max.	Avg.
Redirection	2	12	12	12
Image	2	3	6	4.5
PHP Code	2	1	3	2

Table 5: The number of "malicious" labels per site after applying different obfuscation methods.

pages under three new domain names. Then we choose `NetCraft`, `Forcepoint`, and `Fortinet` which are capable of detecting the PayPal page in the main experiment. We first submit the three URLs to individual vendors for scanning (one URL per vendor). Same as before, the URLs get immediately blacklisted by the respective vendor. Then we submit the URLs to VirusTotal for the first scan. VirusTotal returns a "benign" label for all the URLs. After 4 days, we submit the URL to VirusTotal for the second scan. Interestingly, the returned labels are still "benign". This indicates `NetCraft`, `Forcepoint`, and `Fortinet` do not share their blacklists with VirusTotal. Otherwise, the labels should have been "malicious" after the second VirusTotal scan. It is more likely that VirusTotal runs stripped-down versions of the scanners that fail to detect the phishing pages .

Impact of Obfuscation. Obfuscation is used to deliberately make it harder to understand the intent of the website. In this case, the attacker can apply simple changes so that their website still looks like the target website, but the underlying content (*e.g.*, code) becomes harder to analyze. We examine the impact of three obfuscation methods: (1) `Redirection`: we use a URL shortener service to obfuscate the phishing URL. (2) *Image-based Obfuscation:* we take a screenshot of the PayPal website, and use the screenshot as the background image of the phishing site. Then we overlay the login form on top of the image. In this way, the phishing site still looks the same, but the HTML file is dramatically different. (3) *PHP Code Obfuscation:* within the original PHP code, we first replace all user-defined names with random strings (without affecting the functionality). Then we remove all the comments and whitespace, and output encoding in ASCII. For each of the obfuscation methods, we build 2 new PayPal sites (6 sites in total). We submit the URLs to VirusTotal for scan, wait for a week, submit again (to trigger database update), and retrieve the labels.

Table 5 shows the number of malicious labels per site. As a comparison baseline, without obfuscation, the PayPal site in the main experiment (§4) received 12.1 malicious labels on average. This number is calculated based on the first scan of week-2 in the main experiment (instead of the four weeks of result) to be consistent with the setting of the obfuscation experiment. We observe that redirection does not help much. However, image and code-based obfuscations are quite effective — the average number of malicious labels drops from 12.1 to 4.5 and 2 respectively. This suggests that these vendors are still unable to handle simple obfuscation schemes.

Robots.txt. To see the impact of `robots.txt`, we set up 18 new domains where the `robots.txt` disallows crawling. Then we submit these 18 URLs to the 18 vendors' scan APIs. We find that the traffic volumes are still comparable with the previous experiment. The result indicates that most scanners would ignore `robots.txt`.

Detection of Benign Pages. All the experiments so far are focused on phishing pages. A quick follow-up question is how well can VirusTotal detect benign pages. We did a quick experiment by setting up one benign page under a new domain name (a long random string as before). The page is a personal blog, and it does not try to impersonate any other brand. We submit the URL to VirusTotal scan API twice with 3 days apart, and then monitor the label for a month. We find that the labels are always "benign". Given the limited scale of this experiment, it is not yet conclusive about VirusTotal's false positive rate. At least, we show that VirusTotal did not incorrectly label the website as "malicious" just because it has a long random domain name.

6 DISCUSSIONS & OPEN QUESTIONS

Our experiments in §4 and §5 collectively involve 66 (38+28) experimental websites. We show that vendors have an uneven detection performance. In the main experiment, only 15 vendors have detected at least one site. Even the best vendor only detected 26 out of 36 sites. Given that vendors have an uneven capability, their labels should not be treated equally when aggregating their results. In addition, we show the delays of label updating due to the non-proactive "pull" method of VirusTotal. We also illustrate the label inconsistency between VirusTotal scan and the vendors' own scans. As a simple best-practice, we suggest future researchers scanning the URLs twice to obtain the updated labels and cross-checking the labels with the vendors' own APIs.

Limitations. Our experiments have a few limitations. First, the long domain names may affect the detection accuracy. However, we argue that the long domain names actually make the websites look suspicious, and thus make the detection easier. The fact that certain scanners still fail to detect the phishing sites further confirms the deficiency of scanners. Second, the use of "fresh" domain names may also affect the detection performance of vendors, since certain vendors might use "infection vendors" as features (*e.g.*, reports from the victims of a phishing site). In practice, the vendors might perform better on phishing sites that already had victims.

Future Work. During our experiments, we observe interesting phenomena that lead to new open questions. First, the vendors' models perform much better on PayPal pages than on IRS pages. Future work can further investigate the "fairness" of vendors' classifiers regarding their performance on more popular and less popular phishing brands. Second, we observe that some vendors always detect *the same subset* of phishing sites (Table 4). If these vendors indeed fully synchronize their labels, then their labels are essentially redundant information. As such, these vendors should not be treated as independent vendors when aggregating their votes. Future work can further investigate the correlation of results between different vendors. Third, many vendors (*e.g.*, `Kaspersky`, `Bitdefender`, `Fortinet`) also provide API for file scanning to detect malware. File scan can be studied in a similar way, *e.g.*, submitting "ground-truth" malware and benign files to evaluate the quality of labels and the consistency between vendors and VirusTotal.

ACKNOWLEDGEMENT

We would like to thank our shepherd Gianluca Stringhini and the anonymous reviewers for their helpful feedback. This project was supported by NSF grants CNS-1750101 and CNS-1717028.

REFERENCES

[1] Digital ocean. https://www.digitalocean.com/.
[2] Irs login page. https://sa.www4.irs.gov/ola/.
[3] Joe sandbox. https://www.joesecurity.org/.
[4] Jotti's malware scan. https://virusscan.jotti.org/.
[5] Namesilo. https://www.namesilo.com/.
[6] Paypal login page. https://www.paypal.com/us/signin.
[7] Virscan. http://VirSCAN.org.
[8] Virustotal. https://www.virustotal.com/.
[9] Virustotal faq. https://support.virustotal.com/hc/en-us/articles/115002122285-AV-product-on-VirusTotal-detects-a-file-and-its-equivalent-commercial-version-does-not.
[10] Virustotal public api v2.0. https://www.virustotal.com/en/documentation/public-api/.
[11] Virustotal vendors. https://support.virustotal.com/hc/en-us/articles/115002146809-Contributors.
[12] Akhawe, D., and Felt, A. P. Alice in warningland: A large-scale field study of browser security warning effectiveness. In *Proc. of USENIX Security* (2013).
[13] Aonzo, S., Merlo, A., Tavella, G., and Fratantonio, Y. Phishing attacks on modern android. In *Proc. of CCS* (2018).
[14] Ardi, C., and Heidemann, J. Auntietuna: Personalized content-based phishing detection. In *NDSS Usable Security Workshop (USEC)* (2016).
[15] Cai, Z., and Yap, R. H. Inferring the detection logic and evaluating the effectiveness of android anti-virus apps. In *Proc. of CODASPY* (2016).
[16] Catakoglu, O., Balduzzi, M., and Balzarotti, D. Automatic extraction of indicators of compromise for web applications. In *Proc. of WWW* (2016).
[17] Chen, Y., Nadji, Y., Romero-Gómez, R., Antonakakis, M., and Dagon, D. Measuring network reputation in the ad-bidding process. In *Proc. of DIMVA* (2017).
[18] Cheng, B., Ming, J., Fu, J., Peng, G., Chen, T., Zhang, X., and Marion, J.-Y. Towards paving the way for large-scale windows malware analysis: Generic binary unpacking with orders-of-magnitude performance boost. In *Proc. of CCS* (2018).
[19] Dong, Z., Kapadia, A., Blythe, J., and Camp, L. J. Beyond the lock icon: real-time detection of phishing websites using public key certificates. In *Proc. of eCrime* (2015).
[20] Hong, G., Yang, Z., Yang, S., Zhang, L., Nan, Y., Zhang, Z., Yang, M., Zhang, Y., Qian, Z., and Duan, H. How you get shot in the back: A systematical study about cryptojacking in the real world. In *Proc. of CCS* (2018).
[21] Invernizzi, L., Thomas, K., Kapravelos, A., Comanescu, O., Picod, J., and Bursztein, E. Cloak of visibility: Detecting when machines browse a different web. In *Proc. of IEEE S&P* (2016).
[22] Kantchelian, A., Tschantz, M. C., Afroz, S., Miller, B., Shankar, V., Bachwani, R., Joseph, A. D., and Tygar, J. D. Better malware ground truth: Techniques for weighting anti-virus vendor labels. In *Proc. of AISec* (2015).
[23] Kim, D., Kwon, B. J., and Dumitraş, T. Certified malware: Measuring breaches of trust in the windows code-signing pki. In *Proc. of CCS* (2017).
[24] Kim, D., Kwon, B. J., Kozák, K., Gates, C., and DumitraÈŽ, T. The broken shield: Measuring revocation effectiveness in the windows code-signing pki. In *Proc. of USENIX Security* (2018).
[25] Kleitman, S., Law, M. K., and Kay, J. ItâĂŹs the deceiver and the receiver: Individual differences in phishing susceptibility and false positives with item profiling. *PLOS One* (2018).
[26] Korczynski, D., and Yin, H. Capturing malware propagations with code injections and code-reuse attacks. In *Proc. of CCS* (2017).
[27] Kwon, B. J., Mondal, J., Jang, J., Bilge, L., and Dumitraş, T. The dropper effect: Insights into malware distribution with downloader graph analytics. In *Proc. of CCS* (2015).
[28] Lever, C., Kotzias, P., Balzarotti, D., Caballero, J., and Antonakakis, M. A lustrum of malware network communication: Evolution and insights. In *Proc. of IEEE S&P* (2017).
[29] Li, B., Vadrevu, P., Lee, K. H., Perdisci, R., Liu, J., Rahbarinia, B., Li, K., and Antonakakis, M. Jsgraph: Enabling reconstruction of web attacks via efficient tracking of live in-browser javascript executions. In *Proc. of NDSS* (2018).
[30] Miramirkhani, N., Barron, T., Ferdman, M., and Nikiforakis, N. Panning for gold.com: Understanding the dynamics of domain dropcatching. In *Proc. of WWW* (2018).
[31] Neupane, A., Saxena, N., Kuruvilla, K., Georgescu, M., and Kana, R. K. Neural signatures of user-centered security: An fmri study of phishing, and malware warnings. In *Proc. of NDSS* (2014).
[32] Oest, A., Safaei, Y., Doupé, A., Ahn, G., Wardman, B., and Tyers, K. Phishfarm: A scalable framework for measuring the effectiveness of evasion techniques against browser phishing blacklists. In *Proc. of IEEE S&P* (2019).
[33] Oprea, A., Li, Z., Norris, R., and Bowers, K. Made: Security analytics for enterprise threat detection. In *Proc. of ACSAC* (2018).
[34] Peng, P., Xu, C., Quinn, L., Hu, H., Viswanath, B., and Wang, G. What happens after you leak your password: Understanding credential sharing on phishing sites. In *Proc. of AsiaCCS* (2019).
[35] Razaghpanah, A., Nithyanand, R., Vallina-Rodriguez, N., Sundaresan, S., Allman, M., Kreibich, C., and Gill, P. Apps, trackers, privacy, and regulators: A global study of the mobile tracking ecosystem. In *Proc. of NDSS* (2018).
[36] Sarabi, A., and Liu, M. Characterizing the internet host population using deep learning: A universal and lightweight numerical embedding. In *Proc. of IMC* (2018).
[37] Schwartz, E. J., Cohen, C. F., Duggan, M., Gennari, J., Havrilla, J. S., and Hines, C. Using logic programming to recover c++ classes and methods from compiled executables. In *Proc. of CCS* (2018).
[38] Sharif, M., Urakawa, J., Christin, N., Kubota, A., and Yamada, A. Predicting impending exposure to malicious content from user behavior. In *Proc. of CCS* (2018).
[39] Szurdi, J., and Christin, N. Email typosquatting. In *Proc. of IMC* (2017).
[40] Tian, K., Jan, S. T. K., Hu, H., Yao, D., and Wang, G. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Proc. of IMC* (2018).
[41] Wang, H., Liu, Z., Liang, J., Vallina-Rodriguez, N., Guo, Y., Li, L., Tapiador, J., Cao, J., and Xu, G. Beyond google play: A large-scale comparative study of chinese android app markets. In *Proc. of IMC* (2018).
[42] Wang, L., Nappa, A., Caballero, J., Ristenpart, T., and Akella, A. Whowas: A platform for measuring web deployments on iaas clouds. In *Proc. of IMC* (2014).
[43] Whittaker, C., Ryner, B., and Nazif, M. Large-scale automatic classification of phishing pages. In *Proc. of NDSS* (2010).
[44] Wong, M. Y., and Lie, D. Tackling runtime-based obfuscation in android with tiro. In *Proc. of USENIX Security* (2018).
[45] Xu, D., Ming, J., Fu, Y., and Wu, D. Vmhunt: A verifiable approach to partially-virtualized binary code simplification. In *Proc. of CCS* (2018).
[46] Xu, Z., Nappa, A., Baykov, R., Yang, G., Caballero, J., and Gu, G. Autoprobe: Towards automatic active malicious server probing using dynamic binary analysis. In *Proc. of CCS* (2014).
[47] Zuo, C., and Lin, Z. Smartgen: Exposing server urls of mobile apps with selective symbolic execution. In *Proc. of WWW* (2017).

APPENDIX - RESEARCH ETHICS

We want to provide a detailed discussion on the research ethics.

Internet users. We have taken active steps to make sure our experiments do not involve or harm any users. Given that our experiments require hosting public websites, we need to prevent real users from accidentally visiting the experimental websites and revealing sensitive information. First, all the phishing websites use long random string as domain names (50 random characters). It is very unlikely that users would mistype an actual website's domain name in the address bar to reach our websites.

Second, we never advertise the websites other than submitting them to scanners. We have checked popular search engines (Google, Microsoft Bing) by searching keywords such as "PayPal", "IRS", and "tax". We did not find our phishing websites indexed after examining the first 10 pages of search results. It is very unlikely real users would find our website via search engines.

Third, to prevent the servers accidentally storing sensitive user information (*e.g.*, password), we have modified the PayPal and IRS phishing kits when deploying the websites. More specifically, for any HTTP POST requests, the server will automatically parse and discard the data fields without storing the data. Even if a user accidentally submitted sensitive information via the login form, the data would never be stored in the database or logs. Throughout our experiments, we never received such requests. After the experiments, all phishing pages are taken offline immediately. Certain readers may ask, is it possible that the scanners actually recognized that the password was never stored and thus labeled the website as "benign"? We believe this is unlikely because the action happens *internally at the server side* and the server provides no feedback or error messages. The internal action is invisible to the scanners.

From a scanner perspective, the website behaves like a typical phishing site: (1) a PayPal login page is hosted under a server whose domain name is a long random string; (2) the website sends the login password to a web hosting service instead of `paypal.com`.

VirusTotal and its vendors. Once the URLs are submitted, all the scanning is performed automatically without any human involvement. Given the relatively small-scale of the experiments, we expect the impact to these services (*e.g.*, scanning workload) is minimal.

Domain registrar and web hosting services. We have notified them about our experiments before we start, and have received their consent. This allows us to run the experiments without worrying about any interruption from the domain registrar and hosting services. We do notice that 5 of the 36 URLs used in the main experiment are later included by DNS blacklists (after some vendors of VirusTotal flagged them).

We argue that the experiments' benefits overweight the potential risk. A deeper understanding of how VirusTotal and vendors label phishing sites helps to inform better methods for phishing detection, and inspire new research to improve the label quality.