Forensic Science International: Digital Investigation 36 (2021) 301117

Contents lists available at ScienceDirect

Forensic Science International: Digital Investigation

journal homepage: www.elsevier.com/locate/fsidi

Extended Abstract

Phishing detection on tor hidden services

Martin Steinebach*, Sascha Zenglein, Katharina Brandl. *Fraunhofer SIT, Rheinstrasse 75, 64295, Darmstadt, Germany*
E-mail address: martin.steinebach@sit.fraunhofer.de (M. Steinebach).

Keywords
Tor
Phishing
Image hash
Cloning detection

A B S T R A C T

Phishing is the act of impersonating another party to attack a user, usually stealing information or money. In darknets, where participants are usually anonymous, phishing is a huge problem. We describe the current state of phishing in darknets, especially the Tor network. We analyse what techniques attackers can use to impersonate other services as well as develop some metrics to automatically detect phishing pages. Existing solutions against phishing are presented and phishing detection in the clearnet is examined on the transferability to darknets.

H I G H L I G H T S

- Tor phishing detection.
- Site cloning recognition.
- Similarity measures.

* Corresponding author.

E-mail address: martin.steinebach@sit.fraunhofer.de (M. Steinebach).

1. Introduction

Phishing in general is a big problem nowadays. Where software is hardened against attackers, the typical user often is the weak point in the system. Attackers steal money, information and more by impersonating trusted parties. Whether phishing takes place on the clearnet or the darknet, users have to stay alert to prevent attacks. While there are some solutions to this problem for clearnet websites, detecting fake hidden services is complicated due to the nature of darknets like the Tor network. Domains are composed of seemingly random characters and thus hard to recognize and remember for humans. Additionally, security features like SSL certificates are not widely used, partly because it is hard to get a certificate from a central authority while still remaining completely anonymous.

One common approach is impersonating a shop and taking control of the payment. After creating a phishing page, the link has to be distributed, the most common way being link lists. An advantage for attackers is that they can operate anonymously: therefore they can easily create multiple services. Since there are no authorities or obvious connections between separate pages, a single party can impersonate multiple services and create a replacement as soon as it is identified. As cryptocurrency payments are not reversible, money transfers cannot be made undone once executed.

The darknet (or better privacy-preserving networks in this case) is not a domain where only illegal activities happen. There are numerous examples of legal usages of the anonymity provided by the Tor network. There are many drop-boxes for providing confidential information run by news agencies. Establishing a phishing clone could be of interest for everyone fearing disclosure of information. A clever clone could send everything fished to the actual onion service and only filter out information unwanted by the phishing party. This not a theoretical issue: in 2019 a phising website imitating a Tor SecureDrop service by The Guardian was detected.[1] By this clone, SecureDrop codenames which are pseudonyms for follow-up communication have been harvested and a malware app was advertised.

Numerous scientific publications address phishing detection in the clearnet, like (Khonji et al., 2013) (Gutierrez et al., 2018) (Hu et al., 2019) (Oest et al., 2018). Currently, machine learning models like neural networks are used to classify sites (Mohammad et al., 2014) (Tan et al., 2018) (Ferreira et al., 2018) (Jain and Gupta, 2019) (Nagaraj et al., 2018). The algorithms can detect patterns in scam or phishing pages and evaluate different criteria to find duplicates. Those algorithms could be adapted to the Tor network. A problem is that some listed properties have no meaning in the context of hidden services, so the program has less information to work with.

2. Automated phishing detection

The most simple and common way to impersonate another website is to

[1] https://www.bleepingcomputer.com/news/security/phishing-attack-targets-the-guardians-whistleblowing-site.

https://doi.org/10.1016/j.fsidi.2021.301117

clone and alter the content. Altering the page can usually be done automatically by replacing certain patterns like URLs and cryptocurrency addresses. To show the most current version of the attacked website, an attacker can also perform a man in the middle attack. The phishing page then performs a request for every request of the user and the attacker can scan the content in real time and replace for example cryptocurrency addresses with his own before delivering the page. This approach generally results in higher page load times.
In this section we present algorithms and metrics to automatically detect phishing pages. The goal is identify phishing copies of hidden services by analyzing the data provided by them.

Text Based. Documents are delivered as text that is processed by the browser to display the content. This text is usually composed of HTML, CSS and Javascript and describes the content, layout, structure and the functionality of a page. It will be similar for cloned pages and can be used to detect phishing clones.
The most simple and straightforward method to compare two pages for similarity is to check for equality. This produces a binary result: Two pages are either identical or not. If they are identical, they show the same content. Testing can be executed on the raw data that is served to the client, for example by comparison of cryptographic hashes. Another technique is using compression to determine the similarity of two documents, based on the COAV algorithm designed by Halvani et al. (2017). The core goal is authorship verification for documents and thus different, but the idea of using compression to find similarity is still applicable. The compression similarity is calculated by analyzing the compression behavior of the pages. In addition, there are various robust text hashing algorithms (Steinebach et al., 2013) allowing to find almost identical texts by local comparison of passages. Their approach is similar to the image hash algorithm discussed in the next section.

Image Based. One important feature of web pages are images. They are also used on phishing pages to feign authority. Therefore they can be utilized to detect phishing pages. For the comparison algorithm regarding images, all found images are downloaded and saved. For legal and performance reasons, the images used are only processed and stored as hash values. To prevent skewing the results, images are generally processed as sets and thus duplicate images are ignored.
The first approach is using a well known cryptographic hash function like SHA256 to identify images. However, a simple solution to mitigate the detection of phishing sites by a tool that analyses images with cryptographic hash values is to subtly manipulate the image. The cryptographic hash value of said image will be completely different, making detection impossible.
To improve the detection of similar images, a special hash algorithm for images can be used. Hash algorithms specifically designed for images can produce the same hash for images that are very similar to each other. An example for such an algorithm are the algorithms of the PHash library (Klinger and Starkweather, 2010) or the ForBild robust hash (Steinebach et al., 2012). A comparison metric based on image hashing for web sites is based on the percentage of identified images present on both websites.

Address Based. To generate an onion address, a key pair is necessary. The onion address is then generated by calculating the hash of the public key. Thus it is neither possible to freely choose an address nor to imitate an existing one, because the private key is only known by the legitimate owner. But one can generate key pairs until a domain with the desired properties emerges. As a result, services try to generate domains that are as descriptive as possible for the relevant content, e.g. Facebook is available on facebookcorewwwi. onion. It is not possible to control the complete domain string, thus usually only the first few characters make any sense to humans. When creating a phishing clone, an attacker can aim to generate addresses until the prefix is the same as the original and to generate an address that looks as similar as possible. The longer the string of common characters in the address, the higher the probability of it being deliberately similar: each subsequent character is harder to brute-force than the previous. To calculate the similarity of two addresses s_A, the longest common prefix c is divided by the length l_A of the address, which is commonly 16. This similarity will usually be relatively low because of the high computational effort that is necessary to generate similar addresses. Pages will generally not have more than 8 characters in common, which results in a similarity value of $s_A = 0.5$ It should be interpreted accordingly when choosing a threshold.

3. Evaluation

To test the accuracy of the phishing detection metrics, a testing set of services with known clones together with randomly chosen individual services is composed. We manually selected pages from different clone detection pages and the crawled data. Each of these pages is compared to all the others with a reasonable threshold. To obtain the information about onion services, we need to download the pages. To do this, we execute a broad crawl over as many pages as possible with a suitable starting point, in our case http://deepweblinks.org. To find onion links, HTML documents are parsed for link elements as well as using regular expression matching to find all mentioned links. The matching is necessary because onion services and link lists often list links in plain text, which are not found by regular crawlers. Proxy that connects to tor as a SOCKS proxy. We chose Polipo as web proxy. After integrating the web proxy in the framework, onion addresses can be treated like regular domains. The spider stores the text for each page by saving the document for offline use. We crawled over 15,000 web pages for over 4000 individual onion services. On these pages, 35,000 different images have been found.
The algorithm to detect phishing sites that are exact copies of each other is expected to have few false positives, but also a potentially high rate of false negatives for pages that are slightly changed. We analyze the accuracy of the algorithms that check for document similarity and compare the different variants with text substitution or extraction. The evaluation is executed for every possible pairing of onion services. With the 103 different onion service addresses tested on, this results in 5253 $\left(\sum_{1}^{103}\right)$ comparisons. As expected, not many services can be detected by simply searching for perfect clones. However, as seen in Fig. 1, there is no falsely detected duplicates.
For the similarity based on images, we compare the results based on image identification. The images are identified via their hash values. We used a cryptographic hash algorithm SHA256 to uniquely identify images and dhash to find similar images. As seen, in Fig. 1 a low threshold of 0.1 already produces good results. The only pages not detected are pages without images. With a threshold of 0.1 meaning that 90% of all images are hash-identical and present on both websites, the possibility of detecting pages with just a common logo is low for pages with a few images. The robust hash returns the same results, meaning the test pages either do not contain any phishing pages that subtly modified images or that they do not factor in the result.
Next, we analyze only the service names. This is a really simple method that does not need any other data than the names of services. Based on how much of the addresses matches, the similarity value is calculated. A good threshold would be 0.35, which means all addresses where the first 6 characters are equal are seen as clones. Lower thresholds produce too many false positives because the chance that two addresses begin with the same string randomly grows. On the other hand, a higher threshold would not find many duplicates at all, simply due to the fact that it is not possible to easily create similar addresses. As seen in Fig. 1 the results are promising. This approach has a relatively high false negative ratio because of the fact that not all phishing pages make the effort to construct a similar looking address.
Given the results above, the best identifier is the image detection. We now use the results to look at phishing. We define a threshold, and assume all pages above this threshold are phishing clones. Thus we can treat the similarity function as transitive. We can then categorize all pages into groups with their phishing clones. With the sha image detection algorithm to identify images and a threshold of 0.1, we can find 191 groups of cloned pages out of 4210 services Of those, three pages are cloned over 30 times, another three between 20 and 30 times, and five are cloned 10 to 19 times. The most pages are cloned only a few times, with 107 services only having

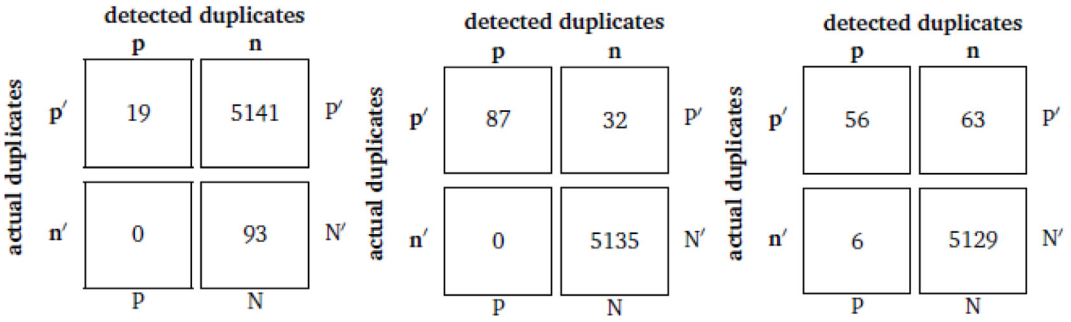

Fig. 1. Confusion matrices for text (left), image (middle) and address (right) comparison.

one clone and the other 73 pages having between 2 and 9 clones. The address based detection gives us a similar result with 112 pages having one clone, 56 with 2–9 clones. However, only 11 pages are in groups with over 10 clones. With the address detection, we see that the results from the image detection have a high accuracy, since pages that have a similar address as well as many shared pages are very likely to actually be phishing clones.

By looking at the results, it is easy to see that there are a few high risk targets. Most of these are Bitcoin Wallets or shops accepting Bitcoin. The anonymous environment and lack of central authorities makes phishing harder to detect for users. Additionally, it is easy to steal money with cryptocurrencies, where addresses can very simply be replaced. Of course, due to the anonymity of the Tor shops, one can not be sure if some or all of the found duplicates are not harmless cases of redundancy. A credible case of redundancy would not be distinguished from malicious clones. Still, due to the observed focus on financial aspects of the Tor net, this seems unlikely.

4. Conclusion

We have shown that it is possible to automatically detect phishing pages with relatively simple techniques. Most phishing pages seem to be generated automatically and can therefore be easily detected. Genuine onion services can take measures to warn their users like announcing the genuine address in a way that is not easily detectable by a computer. However, these methods cannot provide total protection. It is trivial for an attacker to manually create a phishing page that circumvents these detections. Even worse, if a user knows that the page employs special warnings and an attacker imitates, deactivates or hides them, a false sense of security is given. Thus the current solutions are inherently insecure in regards to manually executed phishing.

Acknowledgment

The joint project PANDA on which this publication is based was funded by the Federal Ministry of Education and Research under the funding codes 13N14355 and 13N14356. The authors are responsible for the content of this publication.

References

Ferreira, R.P., Martiniano, A., Napolitano, D., Romero, M., Gatto, D.D.D.O., Farias, E.B. P., Sassi, R.J., 2018. Artificial neural network for websites classification with phishing characteristics. Transactions 3, 6.

Gutierrez, C.N., Kim, T., Della Corte, R., Avery, J., Goldwasser, D., Cinque, M., Bagchi, S. , 2018. Learning from the ones that got away: detecting new forms of phishing attacks. IEEE Trans. Dependable Secure Comput. 15, 988–1001.

Halvani, O., Winter, C., Graner, L., 2017. Authorship Verification Based on Compression-Models arXiv preprint arXiv:1706.00516.

Hu, Z., Chiong, R., Pranata, I., Bao, Y., Lin, Y., 2019. Malicious web domain identification using online credibility and performance data by considering the class imbalance issue. Ind. Manag. Data Syst. 119, 676–696.

Jain, A.K., Gupta, B.B., 2019. A machine learning based approach for phishing detection using hyperlinks information. J. Ambient Intel. Humanized Comput. 10, 2015–2028.

Khonji, M., Iraqi, Y., Jones, A., 2013. Phishing detection: a literature survey. IEEE Commun. Surv. Tutorials 15, 2091–2121.

Klinger, E., Starkweather, D., 2010. pHash–The Open Source Perceptual Hash Library. Technical Report. accessed 2016-05-19.[Online]. Available: http://www.phash. org/apps.

Mohammad, R.M., Thabtah, F., McCluskey, L., 2014. Predicting phishing websites based on self-structuring neural network. Neural Comput. Appl. 25, 443–458.

Nagaraj, K., Bhattacharjee, B., Sridhar, A., 2018. Detection of phishing websites using a novel twofold ensemble model. J. Syst. Inf. Technol. 20, 321–357.

Oest, A., Safei, Y., Doupé, A., Ahn, G.J., Wardman, B., Warner, G., 2018. Inside a phisher's mind: understanding the anti-phishing ecosystem through phishing kit analysis. In: 2018 APWG Symposium on Electronic Crime Research (eCrime). IEEE, pp. 1–12.

Steinebach, M., Klöckner, P., Reimers, N., Wienand, D., Wolf, P., 2013. Robust hash algorithms for text. In: IFIP International Conference on Communications and Multimedia Security. Springer, pp. 135–144.

Steinebach, M., Liu, H., Yannikos, Y., 2012. Forbild: efficient robust image hashing. In: Media Watermarking, Security, and Forensics 2012. International Society for Optics and Photonics, p. 83030O.

Tan, C.L., Chiew, K.L., Musa, N., Ibrahim, D.H.A., 2018. Identifying the most effective feature category in machine learning-based phishing website detection. Int. J. Eng. Technol. 7, 1–6.