POSTER: Probing Tor Hidden Service with Dockers

Jonghyeon Park
Chungnam National University
Daejeon, Republic of Korea
pjh000901@cnu.ac.kr

Youngseok Lee
Chungnam national University
Daejeon, Republic of Korea
lee@cnu.ac.kr

ABSTRACT

Tor is a commonly used anonymous network that provides the hidden services. As the number of hidden services using Tor's anonymous network has been steadily increasing every year, so does the number of services that abuse Tor's anonymity. The existing research on the Tor is mainly focused on Tor's security loopholes and anonymity. As a result, how to collect and analyze the contents of Tor's hidden services is not yet in full swing. In addition, due to the slow access speed of the Tor browser, it is difficult to observe the dynamics of the hidden services. In this work, we present a tool that can monitor the status of hidden services for the analysis of authentic hidden service contents, and have automated our tool with the virtualization software, Docker, to improve the crawling performance. From August 12, 2017 to August 20, 2017, we collected a total of 9,176 sites and analyzed contents for 100 pages.

KEYWORDS

Tor; Deep Web; Dark Web; Hidden Service; Docker;

1 INTRODUCTION

Tor is an anonymous communication tool that uses a special communication method called Onion routing to ensure high anonymity to the user. The high anonymity of Tor applies to web services as a hidden service, and Tor's hidden service has helped people who need freedom and privacy on the Internet. However, the illegal hidden services by abusing Tor are steadily increasing. Because of the anonymity feature of Tor, it is difficult to identify the seller even if illegal goods are traded.

Compared to the increase in these illegal hidden services, previous research has concentrated on the technical aspects of Tor. Tor's classic work has focused on attacking at the network and anonymity [2, 4, 7]. They studied on attack methods using Tor's security defects and factors that can remove anonymity. A research on the contents includes the discovery of the Tor hidden service at the protocol level [4] and it observed the usage of the Tor service through the port scan [2]. This study examined how to discover unknown hidden services using Tor's network weakness or system defects, rather than analyzing the contents of hidden services.

The recent Ransomware incident also uses Tor hidden services for the Bitcoin transaction. A few Tor studies have begun to concentrate on the content analysis and overall scale of hidden service. A study [8] created a special crawler called Dark Crawler. This Dark Crawler was used to analyze the connections between illegal hidden services and the content of extremists and terrorists operating under hidden services at Tor. In [6], they analyzed the content of the hidden service and found the connection point between the normal network and the hidden service through the domain and the resource, and discovered the appearance rate and characteristics of the web tracking through the script used by the hidden service. [1] conducted a two-month intensive analysis of Tor's large illegal marketplace, Agora. They investigated the illegal goods sold in Agora and the country of the seller, and found out the possibility of organized crime. Thus, the recent Tor research is toward the analysis of the content of the large-scale hidden service and the analysis of the ecosystem of the hidden service.

Generally, it is not easy to crawl many Tor hidden services quickly. Hidden service providers open their services for a short time to avoid monitoring and tracking. They occasionally operate most of the hidden services, and often change the address of the hidden service. In addition, due to encryption and peer-to-peer connection of Tor, the access to the Tor hidden service is slow, which is also applied to the Tor crawler. Because of these two features, it is difficult for researchers to collect and analyze the hidden services contents. One the interesting cloud computing environment is to virtualize the computing resource and to automate the job process. As the Tor web browsing is slow, we need a lot of computing resources to collect many Tor hidden service quickly. Due to the popular virtualization software such as Docker, we can easily deploy virtualized computing resources on the cloud. Docker [5] provides containers of abstraction and automation of operating system images, which is contributed to improve the computing resource utilization. Thus, we present a Docker-based Tor hidden service crawler to collect and analyze the hidden services automatically and to monitor them quickly. From experiments, we observed that distributed processing with Docker improves the crawling speed of hidden service pages against Tor's slow communication speed. We also found that a lot of Tor marketplaces and communities are related with illegal trading and Bitcoins.

2 TOR CRAWLER AND ANALYSIS SYSTEM

Figure 1 shows the Tor crawler and analysis system. After we initiate the virtual computing resource on the cloud services such as Microsoft Azure or Amazon EC2, we run one or more Docker container on a cloud computing instance. Each Docker container, a virtualized Linux server, performs as a Tor crawler. To improve the scalability, we load multiple containers of the Docker virtual machine on one instance. We create an image of a dedicated Docker to facilitate configuration and deployment for the crawler. Each Docker container has a set of Tor hidden service addresses to be crawled and it starts to collect Tor hidden services of the whole

CCS '17, October 30-November 3, 2017, Dallas, TX, USA

ACM ISBN 978-1-4503-4946-8/17/10.
https://doi.org/10.1145/3133956.3138849

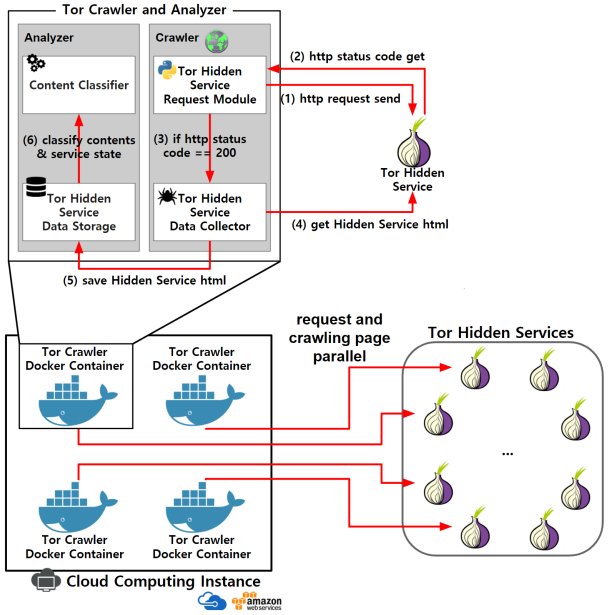

Figure 1: Tor crawling and analysis system architecture

address set. We collected 16,683 Tor hidden service addresses from June to July 2017 with our implementation using an out-of-band hidden service address collection method [3].

We explain the Tor hidden service crawler in each Docker container in Fig. 1. The Tor crawler running on the Docker consists of two processes. One is a Tor hidden service request module that sends a request message, and the other is a Tor hidden service data collector that gathers html files from a hidden service page.

The request module sends a request message to the Tor hidden service server. The request module receives a reply message from the hidden service server and reads the http status code from the reply within a time limit of 30 seconds[1]. If the request is not received, TimeOut or ConnectionError occurs. In this case, we consider the hidden service as dead.

When the request module process finishes, the hidden service data collector checks the http status code and gathers the corresponding html files. We make the Tor crawler collect the hidden service page that actually operates the web page. Thus, the Tor hidden service data collector works only when the http status code is 200. For the address identified as http status code 200, we use the Tor hidden service data collector to access the hidden service page and to save the page's html files. This process, like the request module, has a time limit of 30 seconds.

The Tor analyzer classifies the state of the hidden service measured through the Tor crawler into live or dead states according to the log in Table 1. The Tor crawler actually judges that the hidden service can be accessed in the live state with the http status code of 200. Among the hidden services, there are web pages that request a password for access. These pages return the status code of 401 or 403 because they are not authorized, but the service is operating normally.

[1]30 seconds is the average time when a hidden service request triggers an exception.

Table 1: Tor http return value for classification

Return Value	State	Result
200	Live	html file
401, 403	Live	-
4XX, 5XX	Dead	-
Req or Sel TimeOut/ConnectError	Dead	-

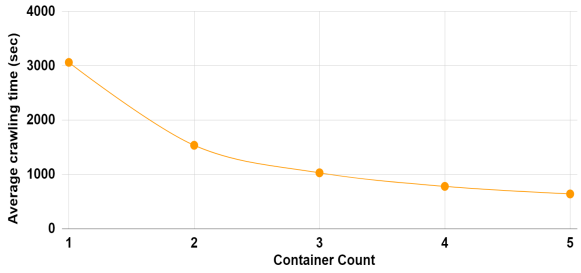

Figure 2: Crawling time by the number of Docker containers.

3 EXPERIMENT RESULT

We have configured five Microsoft Azure instances to test the Tor hidden service's crawling and observing experiment. We used the instance with 2 core and 7GB memory, and Ubuntu 16.04 LST OS. We ran 4 Docker containers on each instance, executing a total of 20 Containers in parallel. The experiment was conducted over 9 days from August 2, 2017 at 2:00 AM UTC to August 20, 17:00 UTC. The number of Onion addresses in this experiment was 16,683, and the addresses were crawled twice a day at 12-hour intervals. Of the total of 20 containers, 11 Docker containers succeeded in collecting data normally. We analyzed 9,176 addresses out of a total of 16,683 addresses.

First, we examine how the virtualization with Docker can improve the Tor hidden service crawling time. To observe the crawling time while the number of containers varies, we select 100 addresses randomly, run each instance from one to five containers, test 100 addresses, and measure the crawling response time. Figure 2 shows the average crawling time as the number of containers increases. For one container, it takes 3,062 seconds to crawl 100 addresses. With five Docker containers, it needs an average of 640 seconds, which is about 4.78 times fast than one container.

Second, we dissect the hidden service contents. We look into the hidden service contents of randomly selected 100 hidden services and extract feature values of words from the collected Tor hidden service. After saving 100 hidden service pages, we classify the contents into 11 categories in Table 2. We classify hidden service contents based on words in the html file. The largest content category is the marketplace of 20%, which is a commodity transaction site for Tor users. In the marketplace, illegal transaction services for prohibited goods such as hacking requests, hitman service, cloned credit cards, and drugs take 16%. In Fig. 3, the upper two sites are the examples of the Tor marketplace hidden service. In the case of the community of 14%, there are communities about the illegal themes such as drug dealers and pedophilia. One unusual category

Table 2: Classification of 100 Tor hidden service samples

Contents Category	Percentage(%)
Marketplace	20
Community	14
Hidden Service directory	12
Personal Blog	9
Onion address sales page	9
Unknown page	9
Bitcoin laundering	6
Wiki	5
Apache default setting page	5
Journalist, Movement organization	4
Software distribution, sales	4
Illegal file sharing	3

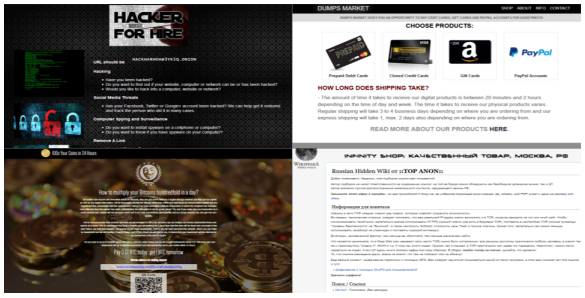

Figure 3: Examples of Tor hidden service pages: hacking request service, cloned credit card market, Bitcoin laundry service and Hidden Service directory

is Bitcoin laundry service of 6%, where users may send Bitcoin with the promotion to receive the returns of 10x to 100x coins. The site at the bottom left of Fig. 3 is a Bitcoin laundry example. Further research is necessary to undermine the usage of the Bitcoin laundry service's Bitcoin transaction. During the collection of these pages, 9% of pages are unknown.

Third, we investigate how Tor hidden services of 9,176 addresses change their states under 18 runs of experiments over 9 days. We crawl Tor hidden services twice a day during period and record the live or dead state of the service. We plot the number of live and dead hidden services for each measurement in Fig. 4. During the observation period, the hidden service averaged 1,166 alive during the whole period and 8,010 services were dead.

8,157 services out of 9,176 were always dead state. We count how often the Tor crawler finds the live state of the target hidden service over 9 days. We observe that 1,019 hidden services were always alive during the observation period. In Fig. 5, 77.2% of the total hidden services were maintained alive at least five days.

4 CONCLUSION AND FUTURE WORK

In this work, we proposed a virtualized Tor crawling and analysis system to monitor the status of Tor hidden services and to classify their contents. We have implemented our tool with Docker on the MS Azure cloud computing platform and have analyzed Tor

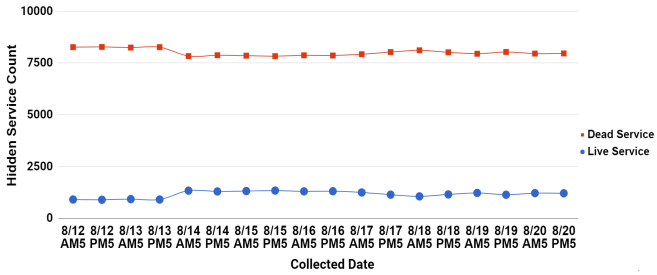

Figure 4: Hidden service state dynamics over 9 days: live or dead

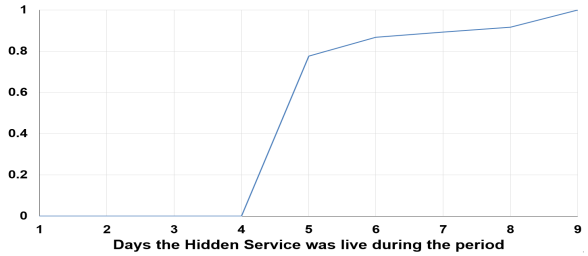

Figure 5: CDF of Tor hidden service live state (day).

hidden services. Based on the results of content analysis, we have shown that illegal Tor hidden services are popular. Our research will contribute to the early discovery of illegal services through scalable Tor hidden service crawling and analysis.

ACKNOWLEDGMENTS

This research was supported by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2017-2016-0-00304) supervised by the IITP(Institute for Information & communications Technology Promotion)

REFERENCES

[1] Andres Baravalle, Mauro Sanchez Lopez, and Sin Wee Lee. 2016. Mining the Dark Web: Drugs and Fake Ids. In *Data Mining Workshops (ICDMW), 2016 IEEE 16th International Conference on*. IEEE, 350–356.
[2] Alex Biryukov, Ivan Pustogarov, Fabrice Thill, and Ralf-Philipp Weinmann. 2014. Content and popularity analysis of Tor hidden services. In *Distributed Computing Systems Workshops (ICDCSW), 2014 IEEE 34th International Conference on*. IEEE, 188–193.
[3] Kang Li, Peipeng Liu, Qingfeng Tan, Jinqiao Shi, Yue Gao, and Xuebin Wang. 2016. Out-of-band discovery and evaluation for tor hidden services. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. ACM, 2057–2062.
[4] Zhen Ling, Junzhou Luo, Kui Wu, and Xinwen Fu. 2013. Protocol-level hidden server discovery. In *INFOCOM, 2013 Proceedings IEEE*. IEEE, 1043–1051.
[5] Dirk Merkel. 2014. Docker: lightweight linux containers for consistent development and deployment. *Linux Journal* 2014, 239 (2014), 2.
[6] Iskander Sanchez-Rola, Davide Balzarotti, and Igor Santos. 2017. The Onions Have Eyes: A Comprehensive Structure and Privacy Analysis of Tor Hidden Services. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1251–1260.
[7] Matthew Thomas and Aziz Mohaisen. 2014. Measuring the Leakage of Onion at the Root. *ACM WPES* (2014).
[8] Ahmed T Zulkarnine, Richard Frank, Bryan Monk, Julianna Mitchell, and Garth Davies. 2016. Surfacing collaborated networks in dark web to find illicit and criminal content. In *Intelligence and Security Informatics (ISI), 2016 IEEE Conference on*. IEEE, 109–114.