A Closer Look At Eclipse Attacks Against Tor Hidden Services

Qingfeng Tan*†‡ Yue Gao*†‡ Jinqiao Shi*† Xuebin Wang*†‡ and Binxing Fang†§
*Institute of Information Engineering Chinese Academy of Sciences, Beijing, China
Email: tanqingfeng@iie.ac.cn
†National Engineering Laboratory for Information Security Technologies, Beijing, China
‡University of Chinese Academy of Sciences, Beijing, China
§Institute of Electronic and Information Engineering in Dongguan UESTC, China

***Abstract*—Tor hidden Services are used to provide anonymity service to users on the Internet without disclosing the location of the servers so as to enable freedom of speech. However, existing Tor hidden services use decentralized architecture making it easier for an adversary to launch DHT-based attacks. In this paper, we present practical Eclipse attacks on Tor hidden services that allow an adversary with an extremely low cost to block arbitrary Tor hidden services. We found that the dominant cost of this attack is IP address resources. The experimental results show that we can eclipse an arbitrary hidden service with 100% success probability with only 6 IP addresses. To understand the severity of the Eclipse attack problems on Tor's hidden services, and its security implications, we present the first formal analysis to evaluate the extent of threat such vulnerabilities may cause and quantify the costs of Eclipse attacks involved in our attack via probabilistic analysis. Theoretical analysis suggests that adversaries with a modest number of IP address resources can block a large number of hidden services at any time.**

Keywords*—Anonymous communications, Tor hidden services, Eclipse attacks, DHT

I. Introduction

Nowadays, with increasing concerns for the government censorship and surveillance against internet users, online privacy and security has drawn increasing public attention. Anonymous communication techniques have become a necessary aim in various Internet applications(e.g., E-voting, web browsing etc). In the past, researchers have designed and implemented numerous anonymous communication systems, including Tor[1], JAP[2] and so on[3]. Tor is a widely used low-latency anonymous communication system allowing Internet users to access network resources without revealing their network address. It current consists of over 7000 volunteer-operated relays, which supports around 2.5 million daily users[4].

In addition to provide sender anonymity, the Tor also provides responder anonymity, called Tor's hidden services, which allow users on the Internet to anonymously host content without revealing the IP address of the servers. The high anonymity for both servers and clients makes Tor's hidden service attractive to Internet users such as political activists fearing Internet censorship and surveillance. The most famous examples include Wikileaks and Globalleaks. Unfortunately, anonymizing services such as Tor's hidden services are facing the issue of abuses for various illegal purposes like drug trading information, child pronography, etc.

Prior research work has proposed various attacks to locate the Tor hidden server, There are main two schemes for an adversary to deanonymize the location of Tor hidden services, either by compromising guard nodes as their entry nodes, or by fingerprinting the underlying network communications so as to identify the hidden service clients and servers. Once revealing physical location of those hidden services, attackers can take down those hidden services which are hosting illegal content, particularly enable identifying the owners. Recent public disclosure by Edward Snowden have demonstrated that the National Security Agency(NSA) is capable of observing and partially deanonymizing Tor hidden services[5]. Among others, The FBI had taken action against over 400 Tor hidden services hosting illegal goods and services for sale on the Tor network[6]. Therefore, Tor's hidden services have been specifically targeted by such adversaries.

In an Eclipse attack [7], an adversary can control the peers' routing table and monopolize all of the victims incoming and outgoing connections, thus the victim will be unable to communicate with the rest of its peers in the network. As Tor is a volunteer-operated overlay network, servers hosting hidden services need to upload its descriptors to hidden service directories(HSDirs) in the Tor network before Tor users will be able to find it. The list of all Tor relays obtaining the HSDir flag form a closed hash ring by the fingerprint size of HSDirs. A hidden service chooses its responsible HSDirs for storing its descriptors by calculating the smallest XOR-distance between its descriptor-ids and the ordered list of HSDir fingerprints. Unfortunately, the descriptor-ids are predictable, and the HSDir fingerprints also can be calculated by an adversary. As a result, the attacker can run conspiring Tor HSDirs and place them in a special position to prevent users from finding the correct hidden services. This design of Tor hidden services that allow an adversary to control look-ups of Tor hidden services that will affect the accessibility of the hidden services, particularly the hidden services will get disconnected from the network.

To understand the severity of the Eclipse attack problems on Tor's hidden services, and its security implications, in this paper, we provide a practical Eclipse attack capable of blocking arbitrary hidden service, and we also develop an analysis framework for evaluating the severity and its security implications against the threat of Eclipse attacks on the live Tor network. We show how relay adversary with modest address resources can always eclipse a targeted hidden service with 100% success rate.

Contributions: In this paper, we take a closer look at the severity and its security implications of the Eclipse attacks on the Tor's hidden services. We start by estimating the amounts of HSDirs controlled by an adversary, additionally, we also present the first formal analysis and attack cost metrics for the threat of this attack in live Tor hidden services. The main contributions of this paper are presented as follow:

1) A practical Eclipse attack capable of disabling arbitrary Tor's hidden services.
2) The first formal analysis framework for evaluating the severity on the live Tor network and its security implications.
3) An in-depth analysis of the costs of this attack, based on actual data of Tor hidden services.

II. Related Work

There have been extensive and ongoing works on Tor hidden services. In this section, we survey the previous researches on attacks against Tor hidden services. Existing attacks on Tor hidden services can be classified into two main categories: deanonymizing Tor'hidden services and denial-of-service attacks.

Deanonymizing Tor hidden services. Researchers have proposed various method for identifying the location of Tor hidden services or hosts. Øverlier and Syverson in[8] proposed fast and cheap attacks that deanonymize hidden services as follows: the adversary deploys a hostile router in the Tor network, then uses a client which repeatedly connect to the target hidden service, and eventually the hidden server will choose the router as its entry guard to the client that allow the attacker to deanonymize the server's IP address by correlating incoming and outgoing traffic. Murdoch and Danezis demonstrated [9] that such traffic correlation attacks be quite efficient against Tor by using some compromised Tor nodes.

Biryukov et al. presented a padding cell method[10], in this attack, the malicious rendezvous point sends a pattern of 50 PADDING cells to the hidden service after receiving a rendezvous1 cell with the attacker's cookie, and then sends a DESTROY cell through the rendezvous to close the circuit, finally, the guard node she controls will be the entry node allowing to monitor the traffics pattern on the circuits to locate the hidden service.

Zhen Ling et al. presented a protocol-level hidden server discovery approach[11]. Kwon et al. proposed a circuit fingerprinting attack[12], which will allow an entry guard to analyze circuit fingerprinting of traffic going through it. In addition, several other attacks have been proposed to deanonymize hidden services [13], these attacks use clock-skew measurement technique through observing the frequency of the system clock changes when repeatedly connecting to a hidden service. Another attack exploits location leaks of Tor's hidden services for deanonymizing Tor hidden services[14], i.e., information in the content or configuration of a hidden service.

Denial-of-Service attack. Distributed anonymity systems, like Tor, I2P, share common failures of DHT-based anonymity schemes[15][16], these security problems stem from attacks on DHT routing, which make it difficult to mitigate by well-known DHT security mechanisms. In previous works, Egger et al.[17] showed a way to perform a classical Eclipse attack in the I2P network by leveraging a group of 20 conspiring nodes to take control over the floodfill database.

Jansen et al. present the Sniper attack[18]. In this attack, an attacker can launch a denial of service attack against Tor by utilizing valid protocol messages to boundlessly consume memory, the Sniper attack can also be used for deanonymizing tor's hidden services through selective denial of service , which will force the Tor client to choose the adversary's guard nodes. Borisov et al. [19] showed that an adversary might carry out selective DoS attacks by choosing only to facilitate connections with other adversary-controlled relays and discard all other connections so that increase their probability of compromising anonymity. Biryukov et al in [10] paid attention to problem of Eclipse attacks that an attacker can complete control over hidden service directories of a particular hidden service, as a result, the attacker can control the access to the descriptors, particularly, this attack will result in services being unavailable.

While prior DoS-based attacks may provide useful information about the Tor'hidden services security problem, they do not tell the details of Eclipse attacks on Tor hidden services, Particularly, these works do not tell how severity the DoS-based attack is. In addition, existing works also don't study the implications of such attacks and quantify the costs of this attack. In order to understand the severity of the Eclipse attack problem, and its security implications, we present a comprehensive analysis framework for evaluating the severity on Tor'hidden services as well as its security implications. By applying this framework, we will be able to obtain quantitative estimates of Tor hidden services's security against DoS attacks.

III. The Eclipse Attack

A. Basic Idea

Observation and Goals. A key observation in the design of Tor's hidden services is that hidden services rely on a security property of the DHT, while the DHT was not designed to defense against eclipse attack. The large DHT with HSDirs run by independent volunteers is essential to security guarantee of hidden services that the descriptor IDs'evolution is entirely random, and not predictable. However, in the current Tor implementation the descriptor IDs are predictable, and the HSDir fingerprints also can be calculated by an adversary.

Follow this basic observation, we present an efficient Eclipse attack against the current Tor's hidden service implementation. This attack works by continuously monopolizing the DHT and blocking all connections to a victim hidden service before clients will be able to contact it. After that, access to that services should be denied for everyone. In this paper, we focus in particular on illegal services like drug trading information, child pronography, etc. that an organization would prefer to see shut down rather than face the issue of abuse for various illegal purposes. Furthermore, we present the severity of the Eclipse attack problem on Tor's hidden services, and its security implications by the formal analysis in the live Tor network.

Our Approach. The key insight behind our approach is to leverage the fact that the descriptor-ids are predictable, and the HSDir fingerprints can also be calculated by an adversary. Therefore, an attacker can calculate the descriptor-ids in advance through finding the appropriate public/private key pairs, in such a way that it is possible to bruteforce the private/public key pairs so that SHA1 hash of the public keys would be in-between the descriptor-id and the fingerprint of the first responsible directory on an arbitrarily date. As a result, the attacker controlled HSDirs can impersonate responsible HSDirs with a high probability. The DoS attack is conducted as follows: (1) Compute the descriptor-ids of the hidden service to determine the responsible HSDir. (2) Find the appropriate public/private key pairs for impersonating the responsible HSDirs, as a result, an attacker can make a relay with an HSDir flag to become a responsible HSDir for a specific hidden service at any given time period. (3) Once an attacker monopolize all responsible HSDirs of a particular hidden service, he can response no data to a client to prevent them from finding introduction points when requesting the targeted hidden service's descriptors, eventually the hidden service will get disconnected from the Tor network.

B. Details of Eclipse Attacks on Tor Hidden Services

Tor's hidden services are built on top of DHT based P2P overlay networks, such DHT-based schemes use a deterministic data placement and ID mapping. This feature on one hand provides assurance on anonymity of location, and on the other hand, facilitates malicious nodes to launch Eclipse attacks that can potentially threaten usability of the system. The process of Eclipse attacks on Tor hidden services can be divided into three phases.

Phase I: Compute the descriptor IDs of the hidden service. Before clients able to contact a hidden service, the hidden service need to pick some introduction points that will be assembled in its descriptors. The descriptors are uploaded to some Tor relays with the HSDir flag. Relays with HSDir flag are hidden service directory servers (HSDirs), which form a distributed hash table, acting as the "DNS server" of Tor network. HSDirs play an important role in the Tor network, because a client has to ask it for introduction points before accessing a hidden service.

Which HSDir can be chosen as responsible HSDir for a hidden service to store descriptors is based on the hidden service's descriptor IDs and the HSDir's fingerprints. In order to launch an Eclipse attack, an adversary need to control the responsible HSDirs of the hidden service, thus it requires to calculate the descriptor-ids of a specific onion address at an arbitrary time. The descriptors are determined by the onion address of the hidden service, time, replica and descriptor-cookie. The replica is used to create different descriptor identifiers, usual from 0 to 1. The descriptor-cookie is typically empty. The algorithm 1 described below can be used to generate descriptor IDs for a targeted hidden service so that they'll select 6 HSDirs as responsible hidden service directories. Hidden services' descriptors will be published to two sets of 3 HSDirs. And descriptor identifiers change periodically every 24 hours.

Phase II: Monopolize hidden service directories and launch an eclipse attack. A hidden service determines if a hidden services directory is responsible based on the descriptor's ID and the directory's fingerprint. Phase I shows that

Algorithm 1: Descriptor IDs Generation Algorithm

```
Input: $onion\text{-}address$, $time$,
$replica$, an integer, currently either 0 or 1,
$descriptor\text{-}cookie$ is an optional secret password of 128
bits that is shared between the hidden service provider
Output: $descriptor\text{-}id$
1 $time\text{-}period$ = (($time$ + first byte of $onion\text{-}address$ *
  86400) / 256) / 86400 ;
2 $secret\text{-}id$ = H($time\text{-}period$ || $descriptor\text{-}cookie$ ||
  $replica$)
3 $descriptor\text{-}id$ = H($onion\text{-}address$ || $secret\text{-}id$);
```

just like any Tor clients, an attacker is able to compute the descriptor IDs of hidden service for any moment, so he can find the expected fingerprints to be responsible directories. The attacker picks appropriate public/private key pairs so that the HSDirs controlled by attacker will be the responsible directory for the hidden service. The algorithm 2 listed below can be used to generate responsible HSDir's fingerprint and its corresponding private key.

Algorithm 2: Fingerprint and Private key Generation Algorithm

```
Input: $descriptor\text{-}id$, $distance$, the distance between
       descriptor-id and its first successor's HSDir
       fingerprint
Output: $fingerprint$, $private\_key$
1 $fingerprint := \emptyset$;
2 while $fingerprint \neq \emptyset$ do
3     $key :=$ rsa.GenerateKey();
4     $id :=$ key.sha1();
5     if $id <$ ($descriptor\text{-}id$ + $distance$) and $id >$
      $descriptor\text{-}id$ then
6         $fingerprint := id$;
7         $private\_key := key$
8 return $fingerprint$, $private\_key$ ;
```

An attacker need to place the private key in a file called secret id key in the keys directory of the DataDir and restart Tor instance, it should be running with expected fingerprint. Then the attacker runs these Tor relays for 96 hours until they obtain the HSDir flag. When the attackers relays appear in the consensus as hidden service directories, they will be used by the hidden service to upload the descriptors and by clients to download the descriptors. In this way, the attacker obtains control over all the responsible directories for a particular hidden service. The attacker can modify the Tor's source code before running them to make the directories return no data to a client requesting the targeted hidden service's descriptor ,thus there would be a complete denial of service.

Phase III: Make the attack persistent. In order to make this Eclipse attack persistent, an adversary requires to impersonate HSDirs of the hidden service with a high degree of covertness over a long period of time. An attacker may try to deny connection probabilistically from Tor users to avoid detection, or choose partial control over the distribution of the descriptors of a hidden service that cause quality of service to degrade. Whereas it's important to determine how many instances the attacker needs to control for a persistent Eclipse attack, preliminary analysis shows that the attacker requires 12 Tor instances for a one-day attack , that is (6 (responsible HSDir) * 2 (24 hour HSDir Delay)) = 12, it is possible to run two Tor instance as HSDir per IP address, therefore attacker needs just 6 static IP addresses.

To make attack persistent, shadowing technique described in [10] is used in the Eclipse attacks, which will require an attacker to set up their nodes for the upcoming descriptor IDs of the targeted hidden service more than 96 hours in advance to make sure they will have received the HSDir flag.

The attacker run 8 Tor instances per static IP, every two Tor instances are a group, 4 groups are running with fingerprints for the upcoming four days after obtaining HSDir flags. For 6 static IPs, at the end of 96 hour time period all of the relays will have HSDir flags but only 12 of them will appear in the consensus and the rest will be shadow relays. The attacker can gradually make active relays unreachable to the Tor authorities according to the time order, so that shadow relays appear in the consensus, thus the attack is persistent. Thus the attacker requires 48 Tor instances for persistent Dos attack, that is (6 (responsible HSDir) * 2 (24 hour HSDir Delay))* 4(4 days for circulation) = 48.

IV. Analysis

In this section, we will present the formal analysis framework for evaluating the severity of Eclipse attacks on the Tor network and its security implications, we attempt to answer the following questions through probabilistic analysis by applying this analysis framework.

1) How serious are Eclipse attacks on Tor's hidden services?
2) What is the probability distribution succeeding in Eclipse attacks for a given hidden service at given time period?

To evaluate the effectiveness of Eclipse attacks on Tor hidden services, we define **attack gap** as the distance between the descriptor-id and its first successor's HSDir fingerprint, and define keyspace size as the total number of all possible fingerprints can be generated. An attacker aims to generate HSDirs' fingerprints until it obtains a few fingerprints within the attack gap for a given hidden services at given time period. Assume attack gap is d and keyspace size is M, thus the probability of a randomly generated fingerprint falling into attack gap is

$$p = \frac{d}{M}.$$

Theorem 1 *In the Tor network, there are n attack gaps we need to monopolize, let $Pr(d, n)$ denote probability mass function that the attack gap size is d:*

$$Pr(d, n) = \frac{n}{M}(1 - \frac{d}{M})^{n-1}.$$

Proof. Let D denote a random variable of the attack gap size with $D \in [0, M)$, This probability that the attack gap size of all hidden services is greater than D is $(1 - \frac{d}{M})^n$, then the cumulative distribution function of the attack gap size is

$$Pr(d < D) = 1 - (1 - \frac{d}{M})^n.$$

consequently, probability mass function of the attack gap size can be derived by

$$Pr(d, n) = \frac{\partial}{\partial d} Pr(d < D) = \frac{n}{M}(1 - \frac{d}{M})^{n-1}.$$

In order to understand the effectiveness of this Eclipse attacks, we need to analyze **Eclipse probability** that an attacker monopolize all responsible HSDirs of a hidden service.

Theorem 2 *If a successful Eclipse attack against a hidden service needs to obtain r desired fingerprints with k attempts, we can define the* ***Eclipse probability*** *as the probability of succeeding in k attempts, thus the probability mass function of K can be calculated by*

$$Pr(K = k|p, r) = \binom{k-1}{r-1} p^r (1-p)^{k-r},$$

where $p = \frac{d}{M}$.

Proof. Since the generation of fingerprints is a sequence of independent Bernoulli trials. In each trial the success probability of obtaining a fingerprint within the attack gap is p, and otherwise is $(1-p)$. In this case, we are observing this sequence until a predefined number r of success has occurred, thus we can define the random variable K as the total number of attempts, which will have the negative binomial (or Pascal) distribution:

$$Pr(K = k|p, r) = \binom{k-1}{r-1} p^r (1-p)^{k-r},$$

where $k = r$, $r + 1$, ...

Theorem 3 *In the Tor network, let K denote the number of attempts to obtain r desired fingerprints with success probability p in an Eclipse attack, the expected number of attempts K can be calculated by*

$$E(K) = \frac{r}{p}.$$

Proof.

$$E(K) = \sum_{k=1}^{\infty} k * Pr(K = k|p, r) = \sum_{k=1}^{\infty} k * \binom{k-1}{r-1} p^r (1-p)^{k-r} = \frac{r}{p}$$

Hence, Theorem 3 shows the mean **computational costs** for varying attack gap size.

Given a hidden service, we want to know what affects the computational costs, this is important in practice since we may not have so many computing power to generate desired fingerprints. Corollary 1 answers this question.

Corollary 1 *The expected number of attempts are inversely proportional to the size of attack gap.*

$$E(K_1|p_1, r) > E(K_2|p_2, r),$$

where $p_1 < p_2$.

Proof. According to Theorem 3, we consider r, p as a constant, thus $E(K_1) = \frac{r}{p_1}$, $E(K_2) = \frac{r}{p_2}$, since $p_1 < p_2$, we have

$\frac{r}{p_1} > \frac{r}{p_2}$, Hence, $E(K_1) > E(K_2)$.

V. Evaluation

We have presented the Eclipse attacks approach in the section III. In order to demonstrate its feasibility, we have implemented a prototype of Eclipse attack approach. In addition, this section also measures the effectiveness of our Eclipse attacks and quantifies the costs based on the probabilistic analysis framework proposed in section IV.

A. Experiment setup

In order to accomplish a practical Eclipse attack, we require to modify existing source code from the Tor project. Then, we calculate the descriptor-ids for given hidden services. When provided with a Tor hidden service onion address, we should output the predicted descriptor-ids which will be used to publish descriptor for this hidden service in the future. The algorithm to generation descriptor-ids has stated in the section III, which can be easily implemented using python language. To get the HSDir flag, it will take at least 96 hours, thus we calculate the descriptor-ids for given hidden service 4 days later. After calculating the descriptor-ids, we use the algorithm 2 to generate responsible HSDir's fingerprint and its corresponding private key.

To demonstrate effectiveness of the Eclipse attacks on the live Tor network, we randomly choose a hidden service as our experiment in October 2015, and then run 6 Tor relays as responsible HSDirs of the hidden service based on the modified version Tor 0.2.5.10. To make the attack persistent, we run 12 Tor instances in 6 static IPs for one attack against one hidden service. However the attack can only continue at most for 24 hours. For the next 24 hours, new descriptor-ids of the hidden service are created, thus the HSDirs with fixed fingerprint cannot receive the descriptors any more. We run 8 Tor instances per static IP, every two Tor instances are a group, 4 groups are running with fingerprints for the upcoming 96h, 120h, 144h, 168h. It is known that it is possible to have two Tor HS directories per IP according to tor specification. What we should explain is that having two Tor hidden service directories does not mean we can only run two Tor instances per IP address, we can run as many Tor instances as possible, but only two Tor instances will appear in the consensus. For 6 static IPs, at the end of 96 hours time period all of the relays will have HSDir flags, but only 12 of them will appear in the consensus and the rest will be shadow relays. We gradually make active relays unreachable to the Tor authorities according to the time order, so that shadow relays appear in the consensus, which make the attack persistent.

B. Experiment results

In this section, we will discuss the experimental results based on the formal analysis and practical experiments of Eclipse attacks.

In order to investigate how the costs of HSDir fingerprints generation could be took, we need to estimate the distribution

TABLE I: Comparing the computational costs with a given number of initial characters.

Number of initial characters	Empirical time(seconds)	Empirical attempts	Theoretical attempts
1	0.18	33	32
2	0.2	862	1024
3	0.32	30282	32768
4	5.18	1119543	1048576
5	21	25833916	33554432
6	840	648685121	1073741824

of gap size in live Tor network, the practical attack gap can be approximately obtained by historical statistics. We crawled many famous websites that contain abundant addresses of hidden services in October 2015, i.e., ahmia.fi, thehiddenwiki.org. Finally, we obtained 3399 effective hidden services, and then computed the attack gap size for these hidden services. Fig. 1 shows the experimental and theoretical cumulative distribution function of the attack gap size under log_{10} scale. These results suggest that the vast majority of hidden services have 2 or 3 same initial characters with their first responsible HSDirs, and the maximum length of same initial characters is only 5. To estimate computational costs, we ran 20 group trials for each different numbers of initial characters on Intel(R) Xeon(R) CPU 2.27GHz computer, as summarized in Table 1. These results show that obtaining a desired fingerprint takes 21 seconds on average in worst case(5 same initial characters), nevertheless, it only takes 0.32 seconds in typical case.

Now estimate the success probability of Eclipse attacks, where each hidden service has different attack gap. We choose the minimal distance between descriptor-id and its first successor's HSDir fingerprint as the attack gap. According to Fig. 1, we find that the minimum attack gap size is 32^{27}, which shows the maximum length of same initial characters between the descriptor-id and its first successor's HSDir fingerprint is 5. Let attack gap is 32^{27} and keyspace size is 32^{32} , applying the Theorem 2, we find the probability of succeeding in k attempts is $\binom{k-1}{5}(\frac{32^{27}}{32^{32}})^5(1-\frac{32^{27}}{32^{32}})^{k-5}$. Fig. 2 and Fig. 3 illustrate the success probability of obtaining a desired fingerprint with the number of attempts. Our experiments suggest that a successful Eclipse attack would require about 25833916 attempts to generate desired fingerprints in worst case, but in typical case, it will take around 30282 attempts.

Based on the formal analysis and practical experiments, we can extrapolate the address resources cost of Eclipse attacks given various hidden services. Since each effective HSDir needs a unique IP address and port combination, and two of the active relays per IP appear in the consensus, which can become the HSDir of the hidden service, this mechanism constraints restrict attackers to 2 concurrent HSDirs(2 fingerprints) on each IP address. However, the active relays not always get HSDir flag simultaneously, thus we need 6 IP addresses against each hidden service to make the attack persistent.

In order to evaluate the severity on the live Tor network, we use the hidden services fraction controlled by adversaries as our security metrics. We collect 3399 hidden services depicted in above. These hidden services fall into 2400 zones on the closed fingerprint circle, which form 2400 attack gaps. If each attack gap can eclipse on average c hidden services and there are n attack gaps we have successfully monopolized, then the

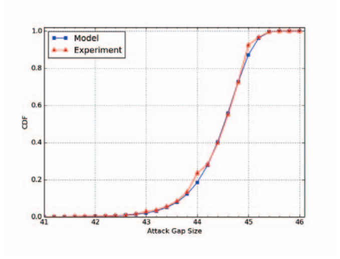

Fig. 1: The CDF of the attack gap size, $\log_{10}$ scale

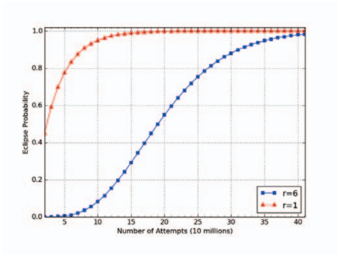

Fig. 2: The probability of succeeding in k attempts in worst case

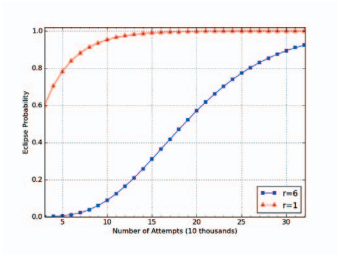

Fig. 3: The probability of succeeding in k attempts in typical case

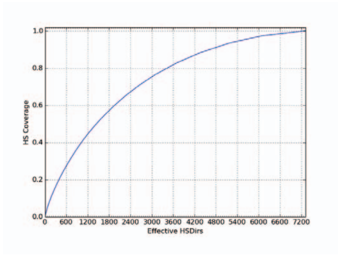

Fig. 4: Hidden services coverage vs. effective HSDirs

attack together will eclipse cn, we can calculate the fraction of the entire hidden services eclipsed by attackers: $s = \frac{cn}{N}$. In practice, the size of attack gap is non uniform distribution(see Fig. 1), and the number of hidden services in each zone don't have a uniform distribution. Fig. 4 illustrates hidden service coverage in the Tor network that we would have controlled using effective HSDirs. This graph shows that we would need 390 effective fingerprints(HSDirs) to eclipse 20% of hidden service, 3420 effective fingerprints to eclipse 80%. Operating 390 HSDirs will require more than 195 IP addresses, and obtaining 390 effective fingerprints only take 390*0.32= 124.8 seconds in the typical case. Thus the experiment shows that the dominant costs are the number of address resources(ie., machines to run Tor instances as HSDirs).

VI. Conclusion

In this paper, we present a practical Eclipse attack and comprehensive analysis framework for evaluating the severity on the live Tor network and its security implications. To demonstrate feasibility of Eclipse attacks, we have implemented a prototype of Eclipse attacks approach on live Tor network. We formalize the Eclipse attacks process as balls-into-bins problem for quantitative estimates of Tor hidden services' security, Our approach defines security metrics with respect to adversaries that show how many IP address resources the attacker need to control for persistent Eclipse attacks and how likely it is that the adversary controls the responsible HSDirs at arbitrary given time period. The results show that Tor'hidden service faces serious risks from Eclipse attacks. In the future, we will discuss potential scheme to mitigate Eclipse attacks on Tor hidden services.

Acknowledgment

This work is supported by Strategic Priority Research Program of the Chinese Academy of Sciences (Grant No. XDA06030200) and Industry-University-Research Cooperation Project of Guangdong Province (2016B090921001).

References

[1] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, ser. SSYM'04. Berkeley, CA, USA: USENIX Association, 2004, pp. 21–21.

[2] O. Berthold, H. Federrath, and S. Köpsell, "Web mixes: A system for anonymous and unobservable internet access," in *International Workshop on Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability*. New York, NY, USA: Springer-Verlag New York, Inc., 2001, pp. 115–129.

[3] A. Ruiz-Martínez, "A survey on solutions and main free tools for privacy enhancing web communications," *Journal of network and computer applications*, vol. 35, no. 5, pp. 1473–1492, 2012.

[4] The Tor Project, "Tor Metrics Portal," https://metrics.torproject.org/.

[5] The Guardian, "NSA and GCHQ target Tor network that protects anonymity of web users," http://www.theguardian.com/world/2013/oct/04/nsa-gchq-attack-tor-network-encryption.

[6] Alex Hern, "Operation Onymous may have exposed flaws in Tor, developers reveal," http://www.theguardian.com/technology/2014/nov/11/operation-onymous-flaws-tor.

[7] E. Sit and R. Morris, "Security considerations for peer-to-peer distributed hash tables," *Lecture Notes in Computer Science*, vol. 2429, pp. 261–269, 2002.

[8] L. Overlier and P. Syverson, "Locating hidden servers," in *Security and Privacy, 2006 IEEE Symposium on*, May 2006, pp. 15 pp.–114.

[9] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, ser. SP '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 183–195.

[10] A. Biryukov, I. Pustogarov, and R.-P. Weinmann, "Trawling for tor hidden services: Detection, measurement, deanonymization," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, ser. SP '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 80–94.

[11] Z. Ling, J. Luo, K. Wu, and X. Fu, "Protocol-level hidden server discovery," in *INFOCOM, 2013 Proceedings IEEE*, April 2013, pp. 1043–1051.

[12] A. Kwon, M. AlSabah, D. Lazar, M. Dacier, and S. Devadas, "Circuit fingerprinting attacks: Passive deanonymization of tor hidden services," in *Proceedings of the 24th USENIX Conference on Security Symposium*, ser. SEC'15. Berkeley, CA, USA: USENIX Association, 2015, pp. 287–302.

[13] S. J. Murdoch, "Hot or not: Revealing hidden services by their clock skew," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, ser. CCS '06. New York, NY, USA: ACM, 2006, pp. 27–36.

[14] S. Matic, P. Kotzias, and J. Caballero, "Caronte: Detecting location leaks for deanonymizing tor hidden services," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 1455–1466.

[15] A. Tran, N. Hopper, and Y. Kim, "Hashing it out in public: common failure modes of dht-based anonymity schemes," in *Acm Workshop on Privacy in the Electronic Society*, 2009, pp. 71–80.

[16] P. Mittal and N. Borisov, "Information leaks in structured peer-to-peer anonymous communication systems," *Acm Transactions on Information and System Security*, vol. 15, no. 1, pp. 359–368, 2012.

[17] C. Egger, J. Schlumberger, C. Kruegel, and G. Vigna, "Practical attacks against the i2p network," *Lecture Notes in Computer Science*, vol. 8145, pp. 432–451, 2013.

[18] R. Jansen, F. Tschorsch, A. Johnson, and B. Scheuermann, "The sniper attack: Anonymously deanonymizing and disabling the tor network," DTIC Document, Tech. Rep., 2014.

[19] N. Borisov, G. Danezis, P. Mittal, and P. Tabriz, "Denial of service or denial of security? how attacks on reliability can compromise anonymity," in *Proceedings of the 14th ACM Conference on Computer and Communications Security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 92–102.