2019 IEEE Symposium on Computers and Communications (ISCC)

SignalCookie: Discovering Guard Relays of Hidden Services in Parallel

Muqian Chen[1,2], Xuebin Wang[1,2], Tingwen Liu[1], Jinqiao Shi[1,3], Zelin Yin[1], Binxing Fang[1,4]
1 Institute of Information Engineering, Chinese Academy of Sciences
2 School of Cyber Security, University of Chinese Academy of Sciences
3 Beijing University of Posts and Telecommunications
4 Electronic and Information Engineering of UESTC in Guangdong
Email: wangxuebin@iie.ac.cn

***Abstract*—In this paper, we propose the SignalCookie attack which can reveal guard relays of multiple hidden services in parallel. The key insight of our method is utilizing Rendezvous Cookie and circuit watermark to deliver the hidden services' identifiers to our controlled relays. By conducting the attack in parallel, the speed of our method increases about 11.6 times compared with the previous work, so that we can continually monitor the guard relays of 13604 hidden services for 7 months. And we have an interesting finding by analyzing the distribution of hidden services binding on each guard relays. The distribution is quite uneven, that only 20% of guard relays serve about 89.32% hidden services. These guard relays would bind all hidden services at least one time in about 17 months. At last, we analyze several security problems aggravated by the uneven distribution, and find that just one corrupt guard relay may cause hundreds of hidden services being de-anonymized or eclipsed.**

***Index Terms*—Tor Hidden Service, Guard Discovery Attack, Guard-HS Relation Analysis**

I. Introduction

Tor hidden service provides a popular way of running an anonymous network service. Besides obfuscating the location of clients, Tor hidden service also provides anonymity for servers, allowing a web server to obfuscate its network location. However, Tor hidden service has also been abused in censorship circumvention by criminals, who are engaging in drug trafficking, child pornography and so on.

Therefore, previous work has carried out multiple methods to locate hidden services. The first unveiled attack against Tor hidden service is proposed by Overlier [1]. The attacker repeatedly requests the target hidden service, and confirms whether the controlled relays are on HS-RP[1] circuit by traffic correlation. At that point, the controlled relays can expose the hidden services' actual IP addresses. In early 2005, in order to mitigate this attack, Tor introduced guard relays to protect the anonymity, which constrains the first node in a small set of relays for a long time [2]. Consequently, the guard relay, which communicates with hidden services directly, has become the key point of the anonymity of hidden services.

Afterwards, Ling [3] and Biryukov [4] separately proposed methods to discover the guard relays of hidden services based on the traffic signals. The controlled relays confirm whether they are the second hops of the HS-RP circuits through detecting traffic signals (e.g. circuit destruction cell sequences or 50 PADDING cells embedded by rendezvous points). To this end, the previous hop is the guard relay of target hidden service. Due to the fact that the guard relays can be easily compromised, coerced, or surveiled by AS-level attackers to determine the actual IP address of the hidden service, Tor project has listed the guard discovery attack as one of the most important attacks against Tor's security [5] in 2018.

Although existing attacks achieve high accuracy when discovering the guard relay of a hidden service, they are not suitable for the scenario with multiple targets. Existing attacks can only embed HS-irrelevant traffic signals into HS-RP circuits for every hidden service, as the Rend-Points (Rendezvous Point) cannot identify which hidden service creates the HS-RP circuit. As a result, in the case of multiple targets, the controlled relays cannot identify which hidden service creates HS-RP circuit through them by detecting the traffic signals. This may cause several problems when the attacker has multiple target hidden services: First, the attacker can only have one target at the same time, which may lead to a long time cost when the attacker has lots of targets. Second, multiple attackers may be interfered by each other when using the same traffic signal, which may reduce the precision of the attack.

To solve these problems, we propose the SignalCookie attack, which can discover guard relays of multiple hidden services in parallel. We utilize a design flaw that a hidden service allows the Client-OP sending information to the collusive Rend-Point through Rend-Cookie. Combining with the circuit watermark, the Rend-Points have the ability to send HS-relevant signal to each HS-RP circuits, enabling attacks to be mounted in parallel . With the help of our SignalCookie attack, we are the first to analyze the security of hidden services by taking a global view of 13604 online hidden services and their guard relays. Consequently, we discover the uneven distribution that a large number of hidden services choose only a small part of nodes to be their guard relays. As a result, this small part of guard relays has the ability to de-anonymize a large number of hidden services. Moreover, a AS-level attacker of these guard relays can de-anonymize quite many hidden services through traffic analyzing [6]–[8].

The contribution of this paper can be listed as following:

- We propose a novel guard discovery attack, SignalCookie,

[1] The circuit created by a hidden service to the Rendezvous Point.

978-1-7281-2999-0/19/$31.00 ©2019 IEEE

which can discover guard relays for multiple hidden services in parallel. Utilizing the Rend-Cookie and circuit watermark, SignalCookie attack is 11.6 times faster than the previous methods.
- In order to analyze the security of the living Tor network, we have studied 13604 hidden services and their guard relays for 7 months, The result of our analysis shows that only 20% guard relays serve about 89.32% hidden services, which indicates that most of hidden services are quite vulnerable to these small part of guard relays. More seriously, we predict that these 20% guard relays can de-anonymize all hidden services in about 17 months.
- We improved current mitigation of SignalCookie attack of Tor network, and analyzed several security problems aggravated by the uneven distribution, which may de-anonymize or eclipse numerous of hidden services in Tor network.

II. The SignalCookie Attack

In this section, we propose SignalCookie attack, and describe the background, basic idea and details of the attack.

A. Background

Fig. 1 shows the five components of the hidden service: Hidden Server (Hidden Service), Client-OP, Rendezvous Point (Rend-Point), Introduction Point (Intro-Point) and Hidden Service Directory (HSDir).

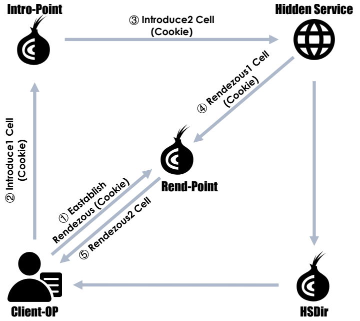

Intro-Point
③ Introduce2 Cell (Cookie)
Hidden Service
② Introduce1 Cell (Cookie)
④ Rendezous1 Cell (Cookie)
Rend-Point
① Eastablish Rendezous (Cookie)
⑤ Rendezous2 Cell
Client-OP
HSDir

Fig. 1. Hidden Service Architecture

The mechanism of the hidden service is described by the previous work extensively [9]. It should be noticed that the Rendezvous Cookie (Rend-Cookie) is an arbitrary 20-byte value, generated randomly by Client-OP. First, it is sent to the Rend-Point by Client-OP and the Rend-Point record the circuit which delivers the Rend-Cookie. Then, the Rend-Cookie is delivered to the hidden service through *introduce1* and *introduce2* cell. When the hidden service receives the designed Rend-Cookie, it will create a HS-RP circuit to the Rend-Point, and send the Rend-Point a *rendezvous1* cell with the Rend-Cookie. Afterwards, the Rend-Point binds up two circuits with the same Rend-Cookie, and delivers messages for these two circuits. At last, the Rend-Point sends the Client-OP a *rendezvous2* cell to start the communication.

B. Basic Idea

In this attack, we assume that the attacker controls a Client-OP, a Rend-Point as well as some Tor relays. Our attack utilizes a design flaw of the hidden service's protocol, that the Client-OP can send messages to the collusion Rend-Point through the Rend-Cookie, because the Rend-Cookie is randomly generated by Client-OP and delivered to the Rend-Point through HS-RP circuit. Our attack embeds the hidden services' identifiers into the Rend-Cookies and delivers them to our Rend-Points. Then our Rend-Points embed the identifiers into the HS-RP circuits as circuit watermarks, so that the controlled relays can identify which hidden service creates HS-RP circuit through them, and achieve to attack multiple hidden services in parallel.

C. Details of SignalCookie Attack

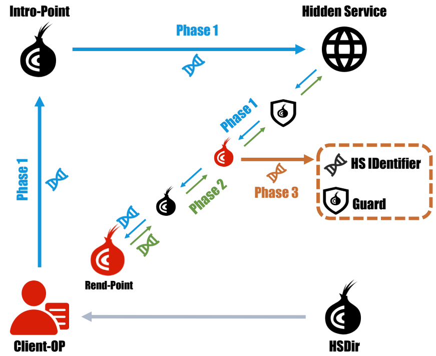


Fig. 2. Details of SignalCookie Attack

The attacker should conduct the attack round by round, until one of its controlled relays is selected as the second hop of the HS-RP circuit. Fig. 2 depicts the three phases of each round of SignalCookie attack.

Phase 1: Rend-Cookie Delivery. Before sending requests to the hidden service, the Client-OP will first generate a designed Rend-Cookie. The designed Rend-Cookie consists of three parts: *Cookie Header*, *HS Identifier* and *Random Content*. *Cookie Header* is a fixed content designed to distinguish the malicious cookies from other common cookies; *HS Identifier* is random value that indicates the identity of target hidden service, which can let the Rend-Point associate the HS-RP circuits with hidden services after receiving Rend-Cookies; And *Random Content* is designed to distinguish the Rend-Cookies with the same target. After generating the Rend-Cookie, each Client-OP picks a controlled relay as the Rend-Point. Then the Client-OP generates a *introduce1* cell including the designed Rend-Cookie and the fingerprint of the selected malicious Rend-Point, and sends the cell to a Intro-Point. The Rend-Cookie will be sent to the selected malicious Rend-Point through Intro-Point and hidden service. When the Rend-Points receive designed Rend-Cookies, they will verify the Cookie Header and extract *HS Identifiers*. As a result, they can identify which hidden service creates the HS-RP circuits to them.

Phase 2: Hidden Service Identifier Modulation. After recognizing the hidden service which created the HS-RP circuit, Rend-Points will send circuit watermarks containing the received *HS Identifiers* along with the HS-RP circuits. Each *HS Identifier* is modulated to the number of *drop* cells[2] in each

[2] *drop* cells are long-range paddings, the OR or OP must drop it when receiving such a cell.

time window (2 seconds). In each window, three *drop* cells represent signal 1, and one *drop* cell is signal 0. For instance, the message 5 ($[101]_2$) can be delivered in 6 seconds as [3 cells, 1 cell, 3 cells]. In particular, we choose this modulating schema because it is easy to be implemented, and it is enough to show the process of the SignalCookie attack. Additionally, the signal modulating schema can be also replaced by a more robust one [10]–[13].

Phase 3: Circuit Watermark Detection. When the controlled relays are selected on the HS-RP circuits, they can recognize the circuit watermarks embedded by Rend-Points. Each controlled relay records the number of received inbound cells in every time window, and generates the cell sequence ordering by time. Next, controlled relays restore the *HS Identifiers* from the cell sequence according to the modulating schema and recognize the hidden services which create the HS-RP circuits through them. Additionally, the number of relay cells before the start of the watermark can be used to figure out their position on HS-RP circuits. The second hop of a HS-RP circuit would receive two outbound relay cells (*extend* cell and *rendezvous1* cell) before the signal starts. Consequently, if a controlled relay is selected on the second hop of the HS-RP circuit, the previous hop will be the guard relay of the target hidden service.

Compared with the previous work, our attack has an extra phase (phase 1), which makes the Rend-Points can identify the hidden services which create HS-RP circuits to them. Hence Rend-Points can embed unique circuit watermarks relevant to hidden services. When one of our controlled relays detects the circuit watermark, it can identify the target hidden service which creates the HS-RP circuit according to the *HS Identifier* in the watermark. Therefore, controlled relays will not be interfered when discovering guard relays of multiple hidden services and the attack can be conducted in parallel.

III. Analysis

In order to make our attack effective, the key insight is that one of our controlled relays should be selected as the second hop of HS-RP circuit. In this part, we analyze the catch probability, which is the probability that a circuit from a hidden service selects one of our controlled relays as the second hop. And the attack cost including resource cost and time cost are also analyzed. At last, we discuss the speed improvement contributed by parallel attacks.

The catch probability is affected by the bandwidth and flags of a relay. A relay's flags are assigned by the directory authorities through voting. According to the relay's flags, we divide Tor relays into four flag types: relays only have guard flag (*guard-only*, g); relays only have exit flag (*exit-only*, e); relays have both guard and exit flags (*guard-exit*, ge); and relays have neither guard nor exit flags (*ge-none*, n). The total bandwidth in Tor network of each type of relays is denoted as B_g, B_e, B_{ge}, B_n respectively.

Suppose that a Tor relay with the bandwidth B' belongs to the type $T \in \{g, e, ge, n\}$, then the probability of the relay selected as the node on the circuit with position $pos \in \{guard, middle, exit\}$ is $P_{pos}(B', T)$. And the $P_{pos}(B', T)$ can be calculated as following [14]:

$$\begin{aligned} P_{pos}(B',T) &= \frac{B' \cdot W^T_{pos}}{B_*} \\ B_* &= B_g \cdot W^g_{pos} + B_e \cdot W^e_{pos} + B_{ge} \cdot W^{ge}_{pos} + B_n \cdot W^n_{pos} \end{aligned} \tag{1}$$

The weights of W^T_{pos} are given in Table I. And the w_g is calculated by $max(0, 1 - \frac{B_g+B_e+B_{ge}+B_n}{3\times(B_g+B_{ge})})$, the w_e is calculated by $max(0, 1 - \frac{B_g+B_e+B_{ge}+B_n}{3\times(B_e+B_{ge})})$, and the w_{ge} is represented by $w_g \cdot w_e$.

TABLE I
Bandwidth Weights

Flag Position	guard-only	exit-only	guard-exit	none
guard	1.0	0.0	w_e	0.0
middle	w_g	w_e	w_{ge}	1.0
exit	0.0	1.0	w_g	0.0

Assuming that the controlled relays which deployed by the attacker have the same bandwidth b, and the numbers of controlled relays with different types are $k_c = \{k_g, k_e, k_{ge}, k_n\}$. So the bandwidth of controlled relays can be represented as $k_c b = \{k_g b, k_e b, k_{ge} b, k_n b\}$.

The catch probability of the controlled relays caught as middle node is $P_m(k_c b)$ (we ignore the increasing of B_* which results from relays deployed by attacker in the following, because $k_c b$ is far less than B_*):

$$P_m(k_c b) = \frac{k_g w_g + k_e w_e + k_{ge} w_{ge} + k_n}{B_*} b \tag{2}$$

Question 1 *What is the relation between the resource cost and time cost of our attack?*

According to the formula (2), the controlled bandwidth and catch probability are proportional, so we can use catch probability to represent the attacker's *resource cost*. Additionally, we consider the *time cost* of our attack as $N_m(k_c b)$, e.g. the number of circuits established before the first time discovering guard relay of the target hidden service. So, the expectation of $N_m(k_c b)$ can be calculated as:

$$\begin{aligned} E(N_m(k_c b)) &= \textstyle\sum_{i=1} i * (1 - P_m(k_c b))^{i-1} P_m(k_c b) \\ &= \tfrac{1}{P_m(k_c b)} \end{aligned} \tag{3}$$

If the attacker creates N circuits, the probability that she discover the guard relay of a target hidden service is:

$$P(N_m(k_c b) < N) = 1 - (1 - P_m(k_c b))^N \tag{4}$$

Question 2 *Compared with the previous work, why the SignalCookie attack improves on speed?*

In order to evaluate the speed of each attack, we use *actual time cost* to express the number of time windows when an attacker discover guard relays of multiple target services. Considering different capability of each hidden services, we define *Service Capability (**C**)* as the max number of circuits created by the hidden service in a time window. Generally, ***C*** reflects the bandwidth of hidden service and the computing power of hidden service's host. Suppose that target hidden services are donated as $\{h_0, h_1, h_2 ... h_n\}$, and the corresponding Service Capability of each hidden service are $\{C_{h_0}, C_{h_1}, C_{h_2} ... C_{h_n}\}$(suppose $C_{h_p} \leq C_{h_q}, p \leq q$). As a result, the *actual time cost* of discovering the guard relay for a hidden services h can be expressed as $\frac{N_m(k_c b)}{C_h}$. Assume that T_1 denotes the *actual time cost* for discovering the guard relay of the all target hidden services $onion = \{h_0, h_1, h_2 ... h_n\}$ using the conventional guard discovery attack, we can calculate the expectation of *actual time cost* as following:

$$E(T_1) \quad = \frac{E(N_m(k_c b))}{C_{h_0}} + \frac{E(N_m(k_c b))}{C_{h_1}} + ... + \frac{E(N_m(k_c b))}{C_{h_n}}$$

The conventional guard discovery attack only permit one target in the Tor network at the same time, so the attacker can just attack the hidden services one by one. However, the SignalCookie attack have the ability to attack hidden services in parallel, so the attacker can approach the Service Capability of all hidden services in theory. So, the T_2, which denotes the *time cost* of SignalCookie attack, can be calculated as following:

$$\begin{aligned} E(T_2) \quad &= max(\frac{N_m(k_c b)}{C_{h_0}}, \frac{N_m(k_c b)}{C_{h_1}}, \frac{N_m(k_c b)}{C_{h_2}}, ..., \frac{N_m(k_c b)}{C_{h_n}}) \\ &= \frac{E(N_m(k_c b))}{min(C_{h_0}, C_{h_1}, C_{h_2}, ..., C_{h_n})} = \frac{E(N_m(k_c b))}{C_{h_0}} < E(T_1) \end{aligned}$$

Obviously, $E(T_2)$ is far less than $E(T_1)$, because the $E(T_2)$ is one of the addends of $E(T_1)$. Consequently, the SignalCookie attack improves the speed significantly when the attacker has multiple target hidden services.

IV. Experiments and The Interesting Finding

In this section, we conduct several experiments of SignalCookie attack both in testing Tor network and living Tor network. In testing Tor network, we evaluate our method in detail. Then, by means of our method, we take a global view of the relations of hidden services and their guard relays in living Tor network, and discover an interesting phenomenon that may affect the security of Tor network.

A. Testing Tor Network Experiment

Experiment Setup: In order to evaluate the cost of our SignalCookie attack, we run the experiments in testing Tor network, which is in a docker-based simulation environment like Shadow [15]. In testing Tor network, we run 10 relays based on a modified Tor (version 0.3.1.7) as the malicious Rend-Points, which can recognize our designed Rend-Cookies and embed the signals into circuits through *relay-drop* cells. Additionally, we run 50 target hidden services and other innocent Tor relays with the original Tor (version 0.3.3.2).

Experiment 1 (Resource and Time Cost): According to our analysis (Section III), resource cost and time cost are related closely to the catch probability. To validate the previous analysis of the catch probability about our attack, we reveal the guard relays of 50 deployed target hidden services through the SignalCookie attack. Before the beginning of each round of attack, we separately deploy $5, 10, 20, 50, 100$ relays as our controlled Tor relays. The bandwidth of each set of controlled relays are denoted as $B_5, B_{10}, B_{20}, B_{50}, B_{100}$, which denotes the different resource cost in each round. Additionally, we use stem [16], a python controller library for Tor, to record each circuit created by the hidden services. Fig. 3(a) depicts the inversely proportional relation between resource cost and time cost, and the practical results comply with our theoretical analysis (Formula (3)). According to the assumption of **Question 1**, the attacker only needs to deploy 8 relays (0.05% bandwidth), so that she can discover the guard relay of hidden service after 2000 requests.

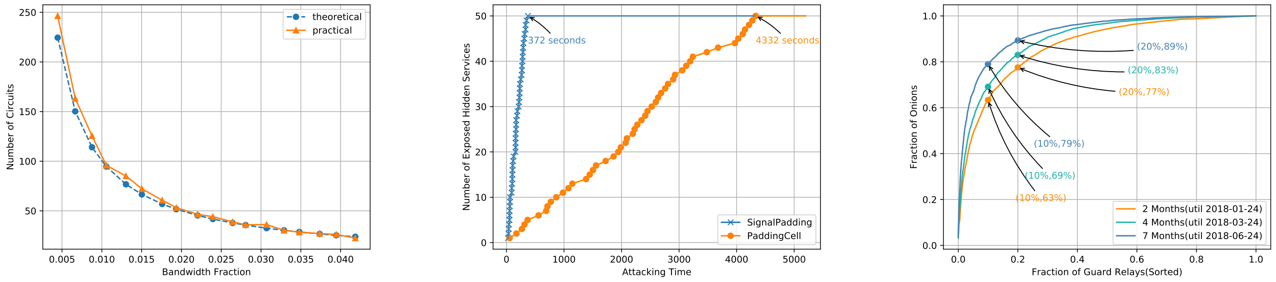
Experiment 2 (Speed Improvement): In order to compare the speed Improvement of our method, we run SignalPadding attack and Biryukov's attack (e.g. PaddingCell attack) and compare the *actual time cost* in testing Tor network. We deploy 10 controlled relays, which are expected to be selected as the second hop of the HS-RP circuit. For both types of attack, we use 50 malicious Client-OPs to attack the 50 deployed hidden services. Firstly we use our SignalCookie attack to discover guard relays of target hidden services, and record the time when we discover the guard relays of each hidden services. Then we attack them through PaddingCell Attack, which is proposed by Biryukov [4], and also record the *actual time cost*. Fig. 3(b) depicts the *actual time cost* of two attacks. Finally, the result indicates that our SignalCookie attack is 11.6 times faster than the PaddingCell Attack.

Experiment 3 (Detection Rate): Along with Experiment 2, we give the detection rate of the process of attacking. The guard relays discovered by these two attacks are all correct, which means that each precision of two attacks are 100%. This is the consequence of the background traffic of testing Tor network is quite clear. However, the recall of PaddingCell Attack is 99.6%, and our SignalCookie Attack has the recall of 93.7%, which means that our attack lost 6.8% signals in the process of attacking. One of the reasons for the loss of these two attacks is the circuit destruction before the signal transmission is complete. Additionally, the signal of SignalCookie may be more sensitive to the delaying or padding, thus the recall is quite higher than the PaddingCell's. This can be prevented by replacing the signal schema with a more robust one, such as the schema proposed in the previous research [10]–[13]. Additionally, Client-OPs can generate a large number of Rend-Cookies with a comparatively low cost, so the loss of signals will never change the result of guard discovery attack.

B. The Interesting Finding in Living Tor Network

The speed improvement of our method enables us to scan all hidden services' guard relays in Tor network. After analysis the result, we find a interesting phenomenon of hidden services and their guard relays, named *uneven distribution*.

Experiment Setup: In living Tor network, we rent 7 Virtual Machines (VM) on cloud environment provided by Vultr. On each VMs, 15 malicious Client-OPs are deployed, designed to generate the malicious requests. We also operate 3 malicious Rend-Points and 10 controlled Tor relays based on the modified Tor (version 0.3.1.7). All of the deployed Tor relays have no Guard or Exit flag, concerning about the ethical issues



(a) (b) (c)

Fig. 3. (a) Number of circuits established by hidden service before selecting controlled relays as the second hop. (b) Improvement on speed of SignalCookie Attack comparing with PaddingCell Attack. (c) CDF of the number of hidden services binding with guard relays.

(Section V-C). We collected 13604 onion addresses of hidden services through the out-of-band discovery [17]. Then we have monitored the guard relays of these hidden services from 11-21-2017 to 06-21-2018, using the deployed Tor relays and malicious Rend-Points.

Experiment 4 (Uneven Distribution): Our Experiment shows the distribution of the binding relations on guard relays is quite uneven. The number of hidden services binding on a guard relay is 20.75 in average. However, about 4.41% guard relays binds more than 100 hidden services. Additionally, Fig. 3(c) depicts that the top 20% of guard relays bind with 77.55% of hidden services in the first two months, and bind with 83.01% of hidden services in four month, and eventually 89.32% of hidden services in seven month.

TABLE II
DISTRIBUTION IN TOP-5 COUNTRIES

Country Name	Guard Relays	Top 20% Guard	Binding HS
German	521 (21.48%)	130 (26.92%)	7889 (57.99%)
France	498 (20.53%)	111 (22.98%)	7666 (56.35%)
America	271 (11.17%)	59 (12.22%)	4249 (32.23%)
Netherlands	212 (8.74%)	48 (9.93%)	3924 (28.84%)
Canada	110 (4.54%)	29 (6.00%)	2290 (7.30%)

It should be noted that, the geographical distribution is also uneven. As shown in Table II, about one fifth of all guard relays locate in German (21.48%) or France (20.53%). Additionally, for the top 5 countries, the proportion of the top 20% guard relays is larger than the proportion of all guard relays. This is the consequence of these countries provide better network infrastructure services than other countries. As a result, German and France has guard relays which bind more than half of the hidden services (57.99% and 56.35%).

In the meantime, the uneven distribution is getting more serious. we model the binding number of hidden services of these 20% guard relays as Coupon Collector Problem [18]. By fitting the percentage of hidden services binding with these 20% guard relays in seven months ($75.13\%, 77.54\%, 79.85\%, 83.01\%, 85.23\%, 87.65\%, 89.31\%$), we predict that these 20% guard relays will de-anonymize all (98%) hidden services in just 17 months.

At last, the uneven distribution would aggravate several security problems of hidden services, just as our discussion in the next section (SectionV-B).

V. DISCUSSION

A. *Mitigation of SignalCookie Attack:*

In July 2018, Tor project proposed an add-on, Vanguards, in order to mitigate the guard discovery attack [19]. Although the add-on can mitigate current guard discovery attack theoretically, it can not provide an inherent defends for Tor users, for the reason that the user needs to download and operate the add-on separately. The add-on has three components: vanguards, bandguards and rendguards. Among of these, rendguards mitigate our SignalCookie attack through enlarging the number of Rend-Points deployed by the attacker. However, after deeply analyzed the mechanism of rendguards, we find that it may cause a high false positive, and we give a more reasonably threshold for the rendguards.

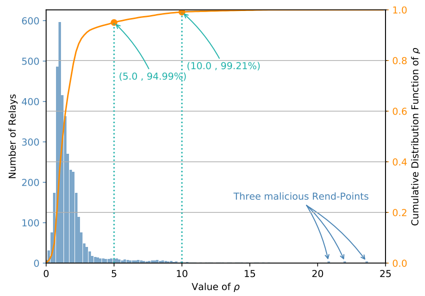


Fig. 4. Distribution of $\rho - value$

For the RendGuard, it is based on the fact that the hidden services can recognize malicious Rend-Points through the chosen frequency of Rend-Point (the frequency of a relay selected as the Rend-Point). The ratio of the chosen frequency and bandwidth for a relay can be denoted as:

$$\rho = \frac{\frac{n_r}{N}}{P_{middle}^{flag}(b)} = \frac{n_r}{N \cdot P_{middle}^{flag}(b)} \tag{5}$$

In this formula, the n_r denotes the time of a relay selected as Rend-Point with the bandwidth b. And the hidden service receives N requests in a certain time window. The threshold of ρ is default set as 5.0, which means that the relay whose ρ is larger than 5.0 may be the malicious Rend-Points.

However, our experiment shows that the threshold may cause a high false positive. To validate this, we deploy a hidden service in the living Tor network and attack it through our SignalCookie attack for 24 hours, we also use normal Client-OPs to request it. The target hidden service records the Rend-Point and the average bandwidth of Rend-Point. Fig. 4 depicts the distribution of the value of ρ for each relay. 94.99% of the ρ-value of relays are lower than 5.0, which means 6.01% requests are failed when Client-OPs request the hidden service using the current RendGuard mechanism. However, 99.21% of the ρ-value of relays are lower than 10.0, and ρ-values of all malicious Rend-Points are higher than 20.0. Consequently, we argue that the threshold of ρ should increase to 10.0 to better distinguish the malicious relays and innocent relays.

B. Security Problems about the Uneven Distribution:

In this section, we discuss two security problems aggregated by the uneven distribution, which may lead to hundreds of hidden services be blocked or de-anonymized in Tor network.

Hidden service de-anonymizing attack: One of the security concerns about the uneven distribution is being de-anonymized by guard relay or malicious ISPs or ASes of a hidden service. On the one hand, guard relays can directly de-anonymize the binding hidden services through our SignalCookie attack. On the other hand, an AS-level attacker of guard relays can easily de-anonymize hidden services by website fingerprinting attack [6]–[8]. As we introduced in Table I, most of top-20% guard relays are in German and France. Each of these two countries has the ability to de-anonymize more than half of the hidden services (57.99% and 56.35%) in this 7 months, this may be a serious problem for the anonymity of hidden services.

Hidden service eclipse attack. Another security problem of aggregated by the uneven distribution is that, one corrupted guard relay can eclipse hundreds of hidden services.

This is based on the fact that a hidden service cannot aware of the dropped cells on a HS-IP[3] circuit. First of all, a guard relay can easily recognize the HS-IP circuits going through it [6], even though it never knows the identity of hidden service. Then, if the guard relay drops all of the incoming cells of a HS-IP circuit, the hidden service cannot be aware of that. Because the HS-IP circuit have no heartbeat cells or outgoing cells after the circuit established, and it cannot trigger the decryption error to detect the dropped cells. Incoming cells (*introduce2* cell) are requests from clients, as a result, requests from clients will never reach the hidden service through the blocked HS-IP circuit. More seriously, a hidden service connects into Tor network just through one guard relay, including all of the HS-IP circuits. Hence, all *introduce2* cells may be dropped by the malicious guard relay, and all of the requests of clients cannot be delivered to the hidden service. This means that the guard relay can monopolize incoming requests of all hidden services binding on it, isolating these hidden services from the users of Tor. Additionally, the eclipse attack sustains util the hidden service changes its guard relay (about 120 days by default).[4]

According to the ***Experiment 4***, the distribution of hidden services and guard relays are quite uneven. The number of hidden services binding on a guard relay is 20.75 in average. However, about 4.41% guard relays binds over one hundred hidden services. As a result, if one of these 4.41% guard relays becomes corrupt, hundreds of hidden services will be eclipsed.

C. Ethical Concerns:

In order to estimate security risk about living Tor users, we conducted our last experiment in living Tor network. However, conducting researches on the living anonymity networks must be performed in a responsible manner. One could be considered as a potential violation of user privacy is the collection of guard relays of hidden services. However, for an attacker without AS-level capability, she cannot track any hidden service's location only through its guard relay. Therefore de-anonymizing hidden services cannot be covered in our study. Additionally, we securely delete all collected data after statistically analyzing them, only publish aggregated statistics about the collected data.

Another factor may also be considered as a potential violation is that, we deploy 10 relays which can record the meta-data for each cell in the living Tor network. However, this is a standard approach in the context of Tor's researches, as it is adopted by the previous work frequently [4], [8], [21]. Additionally, our relays have no Guard or Exit flags, and the meta-data of this kind of relays do not support any of the existing attacks against the anonymity. At last, our experiments are conducted over a period of seven months, and each relays under our control is configured to contribute at least a shared bandwidth of 2Mb/s, which means that we also contribute additional routing capacity to the Tor network.

VI. Related Work

According to the official literature, guard discovery attack is one of the most popular open research topics about Tor's network security [5], as the discovered guard node can be compromised or monitored by an AS-level attacker to determine the actual IP address of the onion service. In the following, we briefly discuss the related work and point out the additional contributions brought by SignalCookie.

The first attack against Tor hidden service is proposed by verlier and Syverson [1]. It based on the fact that a Tor hidden server chooses relays at random to build circuits. The attacker repeatedly connects to the hidden service, and eventually a controlled relay will be the closest one to the hidden server. At that point, by correlating input and output traffic, the attacker can confirm that this is the case, and so he has found the hidden servers IP address. Afterwards, Bauer et al. [22] extended the attack to general purpose circuits (e.g. attacking general Tor users). As the result, guard relays were introduced to protect the anonymity of Tor users [2] by Tor, and become the key point of the anonymity of hidden services.

Afterwards, researchers tend to embed signals into HS-RP circuit in order to confirm whether its controlled relay are on the HS-RP circuit. Biryukov [4] innovated to use

[3] The circuit created by a hidden service to the Introduction Point.

[4] This bug has been reported to the Tor Project by us [20].

50 padding cells as the signal. Once the HS-RP circuit established, a malicious Rend-Point sends 50 padding cells to the hidden service. In the meantime, Ling [3] proposed to use a circuit destruction cell sequence as the signal, that the malicious Rend-Point destroys HS-RP circuit after the circuit being established. The malicious relays confirm whether they are on the HS-RP circuit through detecting their embedded signals. These two method improves the accuracy significantly. However, the attacker can only have one target at the same time, leading to a long time cost when she has lots of targets. Furthermore, multiple attackers may be interfered by each other when they are conducting the attack simultaneously, reducing the precision of the attack.

In 2018, Rochet [23] proposed a side-channel guard discovery attack based on traffic correlation. The attacker sends congestion traffic to the hidden service, and correlates the throughput, which are obtained from the extra-info descriptor, with each guard relays in Tor network. This attack pioneering uses the side channel to get the throughput of guard relays, so that no relays need to be deployed by the attacker. However, this side channel is also revised by Tor project.

One most similar to our method is the Biryukov's work, which is called PaddingCell in our previous sections. Whereas the cornerstone of SignalCookie operation is also sending signals on Rend-Points, the way it does so differs from the related work in two major aspects. First, the Rend-Point, who embeds signals into HS-RP circuit, knows which hidden service are the circuit created from. Second, the signal embedded on the HS-RP circuit is relevant to the hidden service, allowing the attacker has multiple targets in the same time. On top of that, the speed of our attack increases about 11.6 times compared with the previous work. Furthermore, with the help of the speed improvement, we have monitored the guard relays of hidden service in Tor network for 7 months, and discovered several security problems, which is reported to Tor project.

VII. Conclusion

In this paper, we propose the SignalCookie attack that can reveal guard relays of multiple hidden services in parallel. In order to make the attack paralleled, we exploit a design flaw which allows the Rend-Point to identify which hidden service creates HS-RP circuit to it. And then the Rend-Point embeds the identifier into the circuit as circuit watermark. Through detecting the circuit watermark, the controlled relay can recognize the identity of the hidden service that creates the circuit through it, so that our attack can be conducted in parallel. According to our experiments, the speed of our attack increases about 11.6 times compared with the previous work.

With the help of the speed improvement of our method, we take a global view of the relation among 13604 hidden services and their guard relays for 7 months. And we discover the uneven distribution in the living Tor network, that 20% guard relays serve about 89.3% hidden services. Moreover, we predict that these 20% guard relays can de-anonymize all hidden services in about 17 months. For the uneven distribution, we discuss two security problems which may cause numerous of hidden services being de-anonymized or eclipsed. At last, we analyze the mitigation of SignalCookie attack.

Acknowledgments. This work was supported by the National Key Research and Development Program of China (Grant No.2017YFC0821703), DongGuan Innovative Research Team Program (Grant No.201636000100038)

References

[1] L. Overlier and P. F. Syverson, "Locating hidden servers," *ieee symposium on security and privacy*, pp. 100–114, 2006.

[2] M. Wright, M. Adler, B. N. Levine, and C. Shields, "Defending anonymous communications against passive logging attacks," *Proceedings of the IEEE Symposium on Security Privacy*, pp. 28–41, 2003.

[3] Z. Ling, J. Luo, K. Wu, and X. Fu, "Protocol-level hidden server discovery," in *international conference on computer communications*, 2013, Conference Proceedings, pp. 1043–1051.

[4] A. Biryukov, I. Pustogarov, and R.-P. Weinmann, "Trawling for tor hidden services: Detection, measurement, de-anonymization," in *Security and Privacy (SP), 2013 IEEE Symposium on*, Conference Proceedings.

[5] "Tor's open research topics: 2018 edition," Dec 2018. [Online]. Available: https://blog.torproject.org/tors-open-research-topics-2018-edition

[6] A. Kwon, M. Alsabah, D. Lazar, M. Dacier, and S. Devadas, "Circuit fingerprinting attacks: passive de-anonymization of tor hidden services," in *usenix security symposium*, 2015, Conference Proceedings.

[7] R. Overdorf, M. Juarez, G. Acar, R. Greenstadt, and C. Diaz, "How unique is your. onion?: An analysis of the fingerprintability of tor onion services," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 2021–2036.

[8] R. Jansen, M. Juarez, R. Galvez, T. Elahi, and C. Diaz, "Inside job: Applying traffic analysis to measure tor from within," in *Network and Distributed System Security Symposium (NDSS)*, 2018.

[9] "Tor specification." [Online]. Available: https://gitweb.torproject.org/torspec.git/tree/rend-spec-v2.txt

[10] Z. Ling, J. Luo, W. Yu, X. Fu, D. Xuan, and W. Jia, "A new cell-counting-based attack against tor," *IEEE ACM Transactions on Networking*, vol. 20, no. 4, pp. 1245–1261, 2012.

[11] X. Wang, S. Chen, and S. Jajodia, "Network flow watermarking attack on low-latency anonymous communication systems," in *Security and Privacy, 2007. SP '07. IEEE Symposium on*, 2007, pp. 116–130.

[12] A. Houmansadr, N. Kiyavash, and N. Borisov, "Rainbow: A robust and invisible non-blind watermark for network flows." in *NDSS*, 2009.

[13] A. Houmansadr and N. Borisov, "Swirl: A scalable watermark to detect correlated network flows." in *NDSS*, 2011.

[14] N. Danner, S. Defabbia-Kane, D. Krizanc, and M. Liberatore, "Effectiveness and detection of denial-of-service attacks in tor," *ACM Transactions on Information and System Security (TISSEC)*, vol. 15, no. 3, p. 11, 2012.

[15] "Shadow," Dec 2018. [Online]. Available: https://shadow.github.io/

[16] "Stem," Dec 2018. [Online]. Available: https://stem.torproject.org/

[17] K. Li, P. Liu, Q. Tan, J. Shi, Y. Gao, and X. Wang, "Out-of-band discovery and evaluation for tor hidden services," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*. ACM, 2016, Conference Proceedings, pp. 2057–2062.

[18] I. Adler, S. Oren, and S. M. Ross, "The coupon-collector's problem revisited," *Journal of Applied Probability*, vol. 40, 2003.

[19] "Announcing the vanguards add-on for onion services," Dec 2018. [Online]. Available: https://blog.torproject.org/announcing-vanguards-add-onion-services

[20] "Guard eclipse attack," Dec 2018. [Online]. Available: https://trac.torproject.org/projects/tor/ticket/29174

[21] Y. Sun, A. Edmundson, L. Vanbever, and Li, "Raptor: Routing attacks on privacy in tor." in *USENIX Security Symposium*, 2015, pp. 271–286.

[22] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-resource routing attacks against tor," in *Proceedings of the 2007 ACM workshop on Privacy in electronic society*. ACM, 2007, pp. 11–20.

[23] F. Rochet and O. Pereira, "Dropping on the edge: Flexibility and traffic confirmation in onion routing protocols," *Proceedings on Privacy Enhancing Technologies*, vol. 2018, no. 2, pp. 27–46, 2018.