Protecting the Tor network from Sybil attacks

Talk proposal

Philipp Winter
Karlstad University

Roya Ensafi
Princeton University

Karsten Loesing
The Tor Project

Nick Feamster
Princeton University

ABSTRACT

The Tor network is periodically subject to Sybil attacks. Most of these attacks were not carefully executed and Tor directory authority operators were quick to detect and block the malignant relays. However, the Tor network lacks sophisticated tools to detect and protect against Sybil attacks. As a result, more advanced attacks could remain undetected.

While there is a large body of work dedicated to defending against Sybil attacks in peer-to-peer networks, most proposals do not apply to the Tor network as there is no underlying trust relationship between Tor relays. We propose tools to detect Sybils and use them to analyse archived relay descriptors. These tools not only help us gain insights about past attacks and anomalies but they also help us to detect new ones.

Keywords

Tor, Sybil attack, anonymity

1. INTRODUCTION

Table 2 shows that in 2010, somebody set up several hundred Tor relays on PlanetLab machines. In 2012, many clearly related Tor relays,[1] some of which in Amazon's EC2 address space, appeared. In 2014, a small group of relays was suspected to engage in active traffic confirmation attacks [1].

All these attacks were discovered because they were executed carelessly—The Tor Project's Sybil detecting script [2] raised an alert because more than 50 new relays joined the network within an hour. To avoid detection, an attacker could launch a trickling attack, i.e., slowly add Sybils over time rather than all at once. In our ongoing work, we are developing algorithms and practical tools to better detect Sybil attacks, and raise the bar for attackers.

2. SIMILARITY SCORE

We would like to be able to determine the similarity between two given relay descriptors. Our intuition is that similar relays might be run by the same operator. Being able to discover such "relay clusters" would have the following advantages:

- Finding relays that are used for a Sybil attack.
- Once a malicious relay [7] is discovered, we can look for its "partners in crime".
- We can automatically find relays that are supposed to be in the same relay family but their operator failed to correctly configure the `MyFamily` option.
- We can quantify different types of network diversity.

[1]The relay descriptors were highly similar, which made us conclude that the relays were related.

Type	Feature
`bool`	Identical, non-default contact information
`bool`	Identical family information
`bool`	Identical IP address
`bool`	IP address owned by same organisation
`bool`	Identical Tor version
`bool`	Identical exit policy
`bool`	Both relays have directory port set
`int`	Cumulative hours online
`int`	Difference in uptime
`int`	Difference in bandwidth
`int`	Difference in OR port
`int`	Shared fingerprint prefix
`int`	Levenshtein distance between nicknames

Table 1: Fields of similarity vector between two relay descriptors.

We implemented a straightforward similarity score that takes as input two relay descriptors and outputs a heterogeneous similarity vector whose fields are illustrated in Table 1. By linking relay descriptors to additional data such as network consensuses, we can determine more similarities.

We can use this similarity score to determine similarity vectors for all approximately 7,000 relay descriptors in a network consensus. This takes $\mathcal{O}(n^2)$ operations, which is practical for $n = 7,000$ but scales poorly. If we instead use a similarity score that can be used in a metric tree (i.e., the score satisfies the triangle inequality), the complexity can be reduced to $\mathcal{O}(n \log n)$. We experimented with this concept by using the Levenshtein distance between two relay descriptors as similarity metric and vantage point trees as search algorithm [4]. While this approach is significantly faster, it is difficult to incorporate domain-specific knowledge in its similarity metric.

Date	Explanation
2008-05-14	More than 200 relays disappear. Reason yet unclear.
2008-08-19	More than 150 relays disappear. Reason yet unclear.
2008-09-19	A small group called "torism" came online.
2009-09-24	About 60 relays disappear. There's a group of 10 relays with the same nickname, so this might be a false positive.
2010-06-26	Several hundred PlanetLab relays came online. At least their nickname contained "planetlab" or some variation thereof.
2010-09-23	The trotsky relays which were suspected to be part of a botnet.
2010-10-02	Again trotsky relays.
2012-01-23	About 100 relays disappear. Reason yet unclear.
2012-09-16	More than 150 relays disappear. Many of them are in the same /24 and many have the same nickname pattern.
2012-11-15	Several hundred clearly related relays, at least some of which in Amazon's EC2 IP address space, come online.
2013-02-04	A group very similar to the previous one comes online.
2013-04-11	More than 150 relays disappear. Many of them are in the same /24 and many have the same nickname pattern.
2014-01-30	A clearly related group of relays comes online, presumably the one from the pulled Blackhat talk.
2014-04-17	247 relays disappear. These relays were rejected from the consensus because of the heartbleed bug.
2014-04-18	906 relays disappear. These relays were rejected from the consensus because of the heartbleed bug.
2014-11-17	Several probably related relays in the Google cloud get online.
2014-12-26	Many relays named LizardNSA and FuslVZTOR come online.
2014-12-30	Many relays named anonpoke come online.

Table 2: Appearing and disappearing Tor relay clusters over time.

3. PRELIMINARY FINDINGS

Using the tools we are developing (see Section 4), we analysed archived consensuses—ranging back to 2007—and found the following anomalies:

- Many relays changed their fingerprints an unusual amount of times. One Tor relay, 98.212.74.104, changed its fingerprint several hundred times.
- There were undocumented incidents over the past years (see Table 2) in which a suspiciously large amount of relays joined and left the Tor network.

4. ADDITIONAL RESOURCES

We are developing the Go library zoossh [6] to efficiently parse data formats archived by CollecTor [3]. So far, it has partial support for consensuses as well as for server descriptors, and it supports lazy and strict parsing.

We are also developing the Go-based tool sybilhunter [5] that makes use of zoossh to analyse archived data. So far, the following is implemented:

- Determine how often a relay (identified by IP address) changed its fingerprint over time.
- Determine how many relays appear and disappear in two subsequent consensuses.
- Determine the similarity score between all relay descriptors in a given file.
- Use a vantage point tree to find the n nearest neighbours of a given relay.

Finally, all research progress is documented online: `https://www.nymity.ch/sybilhunting/`.

References

[1] Roger Dingledine. *Tor security advisory: "relay early" traffic confirmation attack*. 2014. URL: `https://blog.torproject.org/blog/tor-security-advisory-relay-early-traffic-confirmation-attack`.

[2] Damian Johnson. *doctor*. 2013. URL: `https://gitweb.torproject.org/doctor.git/tree/sybil_checker.py`.

[3] The Tor Project. *CollecTor – Your friendly data-collecting service in the Tor network*. URL: `https://collector.torproject.org`.

[4] Philipp Winter. *Quantifying the similarity between Tor relays*. 2015. URL: `https://lists.torproject.org/pipermail/tor-dev/2015-May/008846.html`.

[5] Philipp Winter. *sybilhunter*. URL: `https://gitweb.torproject.org/user/phw/sybilhunter.git/`.

[6] Philipp Winter. *zoossh*. URL: `https://gitweb.torproject.org/user/phw/zoossh.git/`.

[7] Philipp Winter et al. "Spoiled Onions: Exposing Malicious Tor Exit Relays". In: *PETS*. Springer, 2014.