Preventing Sybil Adversary Attacks on Tor Based Smartphones

Nowsha Islam [#1], Viola Rreza [#2], Sihao Chen [#3], Anuj Devgun [#4], Andersen Chang [#5]

Abstract

Tor enabled smartphones are full featured Android devices with full Tor support, individual application firewalling, true cell network baseband isolation, and optional ZRTP encrypted voice and video support. The phone operates with Orwall, an Android-based firewall that routes network traffic over Tor while blocking all other traffic. These phones use Tor and Tor itself uses onion routers to direct traffic. Communication is encrypted and bounced through a network of nodes run by volunteers around the world; this makes it impossible to determine the sender and receiver of the information sent which is how full anonymity is guaranteed. Access to the network is restricted to an approved list of applications.

Because Tor relays are run by volunteers, it is possible for an adversary to gain control over a large number of relays thus potentially determining the circuit in which the information is being sent. As a result, the anonymity of Tor users would be compromised. Some researchers and the NSA have been observed in the past to use this technique, trying to snoop on some targets.

We aim to create an environment that provides isolated execution which keep sensitive information, including identity keys, safe from the volunteers running the relays. Also, as a preventive measure, a noisy attacker can be easily detected by monitoring scripts like the tor sybil checker which checks to see if there has been a sudden influx of new relays in the system and sends an email.

Introduction

Tor Enabled Smartphones provide more security than an ordinary phone would by allowing anonymous communication. Many security minded people use Tor to protect their right to privacy. In fact, Tor's services are used by more than just the average person who dislikes the idea of companies such as Facebook and Google selling their data to advertisers. Tor allows the military to prevent insurgents from gaining information about military commands and locations of soldiers, activists to report news anonymously without fear of persecution, law enforcement officers to perform sting operations, and many more. Each of these scenarios depend heavily on Tor's ability to guarantee anonymity and the consequences would be dire if this anonymity was compromised.

The key part of Tor that allows its users to remain anonymous is its network of relays. These relays are run by volunteers around the world. However, in a volunteer-based network it is likely that some volunteers are malicious. It is very plausible that an adversary in disguise of a volunteer could gain control of a large number of relays. The adversary could then potentially determine the circuit along which the information is being sent. This would compromise the anonymity of Tor users. A solution to this problem would be to use an environment that provides isolated execution which keeps sensitive information, including identity keys, safe from the volunteers running a relay. Without access to identity keys, volunteers cannot determine a circuit thus protecting the anonymity of Tor users.

Sybil Adversaries have been observed in the past multiple times by the tor community, like the Lizard Squad attack in 2015 that tried to create a large number of nodes, but the community had monitoring in place and hence was able to detect such an attack in time. However, not all attacks are bound to be that

noisy and a more sophisticated sybil attack could still easily get away like the attack NSA used.

Background

When using the Internet, users often inadvertently leave a trail that allows strangers access to personal information about the user. Users are also subject to a form of Internet surveillance known as “traffic analysis.” Traffic analysis involves examining messages to infer who is to talking whom. This is possible because Internet data packets are composed of the data being sent and a header which discloses the source, destination, size, timing, etc. By analyzing the header, the attacker can deduce who is talking to whom, when they are talking and for how long. Encryption does not defend against this kind of attack as encryption only hides the data, not the header. Tor prevents traffic analysis which allows users to anonymously use the Internet.

Tor consists of volunteer-run relays, nine directory servers/authorities, and Tor clients. The directory authorities are dedicated servers run by trusted individuals that tell Tor clients which relays make up the Tor network. Every hour, the directory authorities create a network consensus which contains a summary about all the Tor relays that are currently online. Every relay in the consensus document has a descriptor which contains information about the relay, including information about the relays keys. Each relay has a long term identity key which is used to sign the consensus document and TLS certificates (maybe talk about TLS.) If the identity key of a relay was compromised, an attacker would be able to impersonate the relay and modify its behavior. Our solutions intends to keep the identity key secure and hidden to prevent this impersonation.

Information about all Tor relays is publically available, including their IP addresses. This makes it easier for some governments or ISPs to ban Tor. This is why Tor offers bridges, Tor relays that are not publicly listed as part of Tor network. This makes it harder for governments or ISPs to block them but allows adversaries to stay hidden in the network and perform an attack.

Threat Model

Exit traffic tampering:

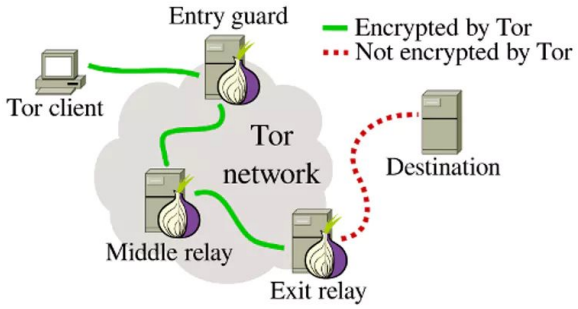

Fig 1. Tor Circuit where jump from exit relay to destination is not encrypted

Even though Tor provides full anonymity for users, the data eventually has to be decrypted before it is sent to the destination. That operation takes place between the exit node and destination when it leaves the Tor network and rejoins the regular internet. Attackers can potentially intercept the network traffic which can be emails, private messages, and personal credentials. Only some data comes with HTTPS and TLS encryption but the best bet is to add an extra layer of security to ensure end-to-end encryption for the data.However, there is still always a possibility of a Man-in-the-middle attack where an eavesdropper pretends to be the receiver during key exchange and makes sender encrypt the data with a key from the eavesdropper.

Website fingerprinting:

Regarding to the entry guard relay, network traffic entering the hop is encrypted

under the Tor network to prevent attackers from gaining access to users' online movement. However, attackers can still figure out what website users visit. Due to the nature of different websites have different packet lengths of data and timings, they can track the traffic flow to predict the identity of each website with a high accuracy rate. Potentially, we can add useless data to increase the packet length to a common length for all websites to prevent attackers from differentiate the websites but it will significantly hamper the network performance and create a lot of overhead.

Bridge Address Harvesting:
The attacker can get bridge address by checking relay connection with the previous relays if it's publicly known or not. A possible solution would be enclosing the Tor network with a trusted environment so attackers need identity keys to get access to the relay traffic. Of course, that requires extra implementations which introduce some sort of overhead even if we are trying to maximize the efficiency.

End-to-end correlation:
By running both entry and exit relays, and analyzing traveling time, an attacker is able to correlate users identity to his/hers destination. Since an attacker needs to run two relays in order to do time analysis, the Tor network can prohibit any volunteer to run more than one relay. However, attackers can potentially use different identity to run two relays.

Design

The proposed solution is a two step method. First, we aim to detect and remove malicious relays from the Tor network. Second, we aim to use a trusted execution environment to mitigate the damage undetected malicious relays can cause.

To detect Sybil relays, we rely on the fact that these relays have similar characteristics and behave similarly. Since Tor aims to be a low latency service, it adapted to route traffic to relays with high bandwidth and high uptime, which is a measure of system reliability. Therefore, malicious relays often have high bandwidth or uptime. It is also possible that the relay only appears to have these characteristics when in fact, they lied about their resources. This is possible if the relay has access to its identity key. In either case, this is done so that more traffic is routed to them. They also go online and offline at the same time, have the same nicknames and port numbers, and have other similar configurations. Using this information, we can run analysis and detection scripts to analyze relays and determine which ones are malicious using pattern recognition in relay names and addresses, which would run in conjunction to the fingerprint check on the site we performed earlier. The malicious relays would then be given a BadExit Flag which tells Tor to avoid exiting through those relays.

Although the above solution will help reduce the number of sybil relays, it is not guaranteed to prevent all sybils from entering the network. The next part of the solution is to use a trusted environment that will limit the what relays can do. A trusted execution environment is an isolated environment that allows isolated execution and confidentiality of the assets of different applications. In this scenario, the descriptors for each relay, and more importantly the identity keys contained by the descriptor, are kept confidential in secure containers. As a result, the volunteers themselves do not have access to the identity keys. Since it is difficult to determine which volunteers are trustworthy, it is necessary to deny all volunteers access to this information.

In onion routing, the data is encrypted along with the next node destination multiple times before it is sent through the virtual circuit of relays chosen at random. With each relay, a layer of encryption is decrypted in order to reveal the next relay that the data will be passed to in the circuit. The final relay decrypts the inner layer of data and sends said data to its destination without revealing or even knowing the source IP address which is how the Tor user maintains his/her anonymity.

In our enclosed Tor network, a trusted overhead component would be in charge of storing the identity keys along with sending the data to the next randomly chosen relay. Essentially, all that the relays do in this new Tor environment is simply ask the overhead for the decrypted data before sending it to another randomly chosen relay. This solution does create a new concern which is the possibility of the adversary eavesdropping on the overhead component. We consider hashing the identity keys to add another layer of security.

Implementation

To implement, we need to check/monitor bad relays behavior and appearance similarity. We might frequently do a minor upgrade of Tor system to check the correlations of several adversaries. When an upgrade occur, every relay has to update, configure and reboot. Bad relays are frequently configured similarly and are managed as a group, which usually goes off line and returns online simultaneously. Also, k-nearest neighbor or other regression method can be used to detect relays which has similar configurations, characteristics etc.

Second, we take away each relays identification key and maintain a trusted execution environment, so even if an adversary gets into traffic, it will not have the ability to decrypt and expose secure information.

By using safe path selection algorithm, we can judge each relay based on their character and reputation and properly choose a high rank or relatively safe relay as bridge. The user can minimize the probability that a bad relay is at both side of entry and exit.

Evaluation

First, since we have to check bad relays behavior, it requires data storage which might affect the network performance. It needs a larger bandwidth to meet standard throughput rate or it will create significant delays to send data over the network.

Second, for a trusted environment, it creates some sort of overhead since we are implementing extra features for the Tor network. The identity keys have to be transferred over the network since we keep that hidden from the volunteer relays. The keys need end-to-end security since an attacker can get access to them if they intercept the network traffic.

To maximize the efficiency of our implementations, we have to minimize the data that are required to transfer over the network.

Related Solutions

There are various papers on how Tor tries to prevent sybil attacks as well as new detection methods that use a combination of IP configuration, time when the node went up as well as monitoring the MAC addresses to detect patterns, but the accuracy for these methods are low, and detection of an attack does not always implement a way of stopping the attack. Some of the methods that have been successful in preventing an attack are The Tor Project's "Sybilhunter" and fingerprint analysis.

Some of the other solutions fall under a broad categorization of interrupting what the

sybils do after hijacking the Tor network, which deals with attacks like Rewriting Sybils, Redirect Sybils, FDCServers Sybils, Botnets and Academic purpose attacks including research.

Conclusion

Detecting for bad adversary requires overhead computation and storage to keep sensitive information. This prevents some attacks but fails to achieve over 90% accuracy on the synthetic tests. Utilizing a trusted execution environment can keep information safe even if a bad relay successfully gets into Tor traffic. By storing all relays reputation and character parameter, it is possible to choose a better path to avoid randomly choosing a bad relay as bridge. Since smartphones have dynamic IPs and move around a lot, it is hard to keep track of the nodes, helping reach the overall goal of anonymizing users given that we are successfully able to mask the device fingerprinting parameters.

References

1) Sybil adversaries in Tor https://nymity.ch/sybilhunting/pdf/sybilhunting-sec16.pdf
2) Low resource attacks on Tor https://nymity.ch/sybilhunting/pdf/Bauer2007a.pdf
3) Malicious nodes in Tor https://nymity.ch/sybilhunting/pdf/Danezis2009a.pdf
4) In September 2007, Dan Egerstad, a Swedish security consultant, revealed he had intercepted usernames and passwords for e-mail accounts by operating and monitoring Tor exit nodes https://www.wired.com/2007/09/rogue-nodes-turn-tor-anonymizer-into-eavesdroppers-paradise/?currentPage=all
5) In October 2011, a research team from ESIEA claimed to have discovered a way to compromise the Tor network by decrypting communication passing over it. https://thehackernews.com/2011/10/tor-anonymizing-network-compromised-by.html
6) Tor identity keys and connection keys https://nymity.ch/anomalous-tor-keys/pdf/anomalous-tor-keys.pdf
7) Types of Tor relays https://trac.torproject.org/projects/tor/wiki/TorRelayGuide
8) Tor Descriptors https://stem.torproject.org/tutorials/mirror_mirror_on_the_wall.html#what-is-a-descriptor
9) Tweet from Tor explaining directory authorities https://twitter.com/torproject/status/926088882705244161?lang=en
10) Tor sybil checker https://gitweb.torproject.org/doctor.git/tree/sybil_checker.py
11) https://taesoo.kim/pubs/2017/kim:sgx-tor.pdf
12) https://www.theregister.co.uk/2016/02/29/tor_takes_aim_against_sybils_on_the_network/