TOR Security: Prohibit DNS Leakage through Torsocks

SUMAN GAUTAM1, NITESH GUPTA2

Abstract: The Tor project provides individuals with a mechanism of communicating anonymously on the Internet. Moreover, Tor is able of providing anonymity to servers, which are configured to get inbound connections only via Tor—more commonly called hidden services. In order to route demand to these hidden services, a namespace is used to verify the resolution requests to such services. A namespace under a non-delegated (pseudo) top-level-domain (TLD) of .onion was elected. Although the Tor system was designed to prevent .onion requests from leaking into the global DNS resolution process, numerous requests are still recognized in the global DNS. Many Tor users don't realize that when they use Tor, there is a many hazard that although connections across Tor network are torrified and encrypted, DNS requests made within the torrified application will not be routed via the Tor network. Therefore, the DNS requests will be made via our own machine. This is a security hazard because anyone watching our computer or our DNS server can tell what names we are looking up, and guess that those are the domains we are visiting via Tor. So, DNS requests not sent through Tor network by default and Attacker could see what websites are being visited. In last paper, the Traffics going over Tor do not encrypted, it’s just anonymous and also used malicious exit node and port or router can observe traffic and identified attacker/hacker. So, in this case, number of ports/routers is injected by attacker. This process takes more time to identify attacker. we overcome these entire problems and provide best solution through Torsocks. Torsocks can prohibit DNS leakage due to things like DNS prefetching or binary plugging that generate their own network traffic, whereas HTTP or SOCKS proxies cannot prohibit these kinds of leaks.

Key words: TOR, Torsocks, DNS leak, proxy, Hidden Services,Torrified,TLD.

Introduction:

TOR is a network of virtual tunnels that allows people and groups to improve their privacy and security on the Internet. It also implement software developers to create new communication tools with built-in security features. Tor provides the authority for a range of applications that allow organizations and individuals to share information over public networks without compromising their security. The term "onion routing" refers to application layers of encryption, nested like the layers of an onion, employ to anonymize connection. Tor encrypts the original data, including the destination internet protocol address, multiple times and sends it via a virtual circuit comprising consecutive, randomly selected Tor relays. Each relay decrypts a layer of encryption to confess only the next relay in the circuit in order to pass the remaining encrypted information on to it. The final relay decrypts the innermost layer of encryption and sends the original information to its destination without acknowledge, or even knowing, the source IP address. Because the routing of the connections is separately concealed at every hop in the Tor circuit, this method excludes any single point at which the communication can be de-

SUMAN GAUTAM, MTECH (CSE), NIIST, Bhopal. Affilied to RGPV, Bhopal, M.P, India Suman.sidhi@gmail.com

NITESH GUPTA, Asst. Professor (CSE) in NIIST, Bhopal. Affiliated to RGPV Bhopal, M.P, India. 9.nitesh@gmail.com

anonymized through network surveillance that relies upon knowing its source and destination.

We need to Tor protects us against a common form of Internet vigilance known as "traffic analysis." Traffic analysis can be employ to infer who is talking to whom over a public network. Knowing the source and destination of our Internet traffic allows others to track our behaviour and interests. This can impact our check book if, for example, an e-commerce site uses price discrimination based on our country or institution of origin. It can even threaten our job and physical safety by revealing who and where we are. For example, if we are travelling abroad and we connect to our employer's computers to check or send mail, we can inadvertently reveal our national origin and professional affiliation to anyone observing the network, even if the connection is encrypted. Even if we encrypt the data payload of our communications, traffic analysis still reveals a great task about what we are doing and, possibly, what we are saying. That's why it focuses on the header, which reveal source, destination, size, timing, and so on. Generally, the problem arise for the privacy/security minded is that the recipient of our connections can see that we sent it by looking at headers. So can authorized negotiator like Internet service provider (ISP) and sometimes unauthorized negotiators as well as a very simple form of traffic investigation might involve sitting somewhere between sender and recipient on the network, looking at headers. But there are more impressive types of traffic analysis. Some attackers spy on multiple sector of the Internet and employ sophisticated statistical techniques to track the connections patterns of many different organizations and individuals. Encryption does not support against these attackers, since it only shield the content of Internet traffic but never shield the headers.

To overcome these problem, Tor helps to reduce the risks of both simple and sophisticated traffic analysis by distributing our transactions over several places on the Internet, so no single point can link us to our destination. The plan is similar to using a twisty, hard-to-follow path in order to throw off somebody who is tailing us and then periodically erasing our footprints. Instead of taking a direct path from source to destination, data packets on the Tor network choose a random paths via several relays that cover our tracks so no observer at any single point can tell where the data came from or where it's going.

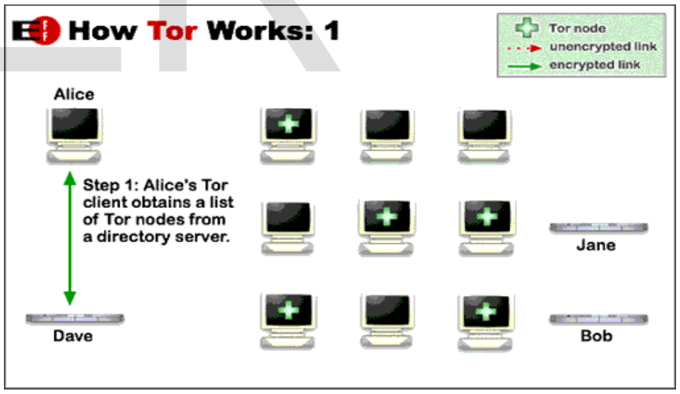
How Tor Works: 1
Tor node
unencrypted link
encrypted link
Alice
Step 1: Alice's Tor client obtains a list of Tor nodes from a directory server.
Jane
Dave
Bob

Figure 1: TOR WORKING

To create a private network paths with Tor, the user's software or client incrementally frame a circuit of encrypted connections through carry on the network. At the time the circuit is extended one hop , and each relay along the way knows only which relay gave it information and which relay it is giving information. No individual relay ever knows the complete routs that a data packet has taken. The client negotiates a separate part of encryption keys for each hop along the circuit to assure that each hop can't trace these connections as they pass through.

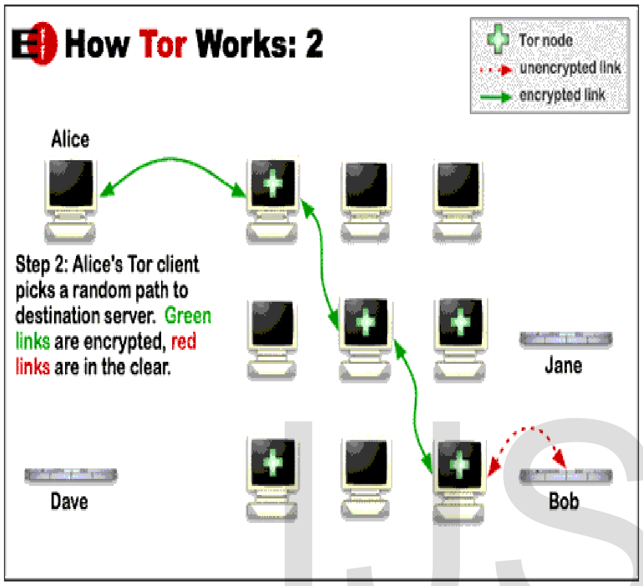
How Tor Works: 2
Tor node
unencrypted link
encrypted link
Alice
Step 2: Alice's Tor client picks a random path to destination server. Green links are encrypted, red links are in the clear.
Jane
Dave
Bob

Figure 2: TOR WORKING

Once a circuit has been established, many types of data can be exchanged and several different sorts of software applications can be expand over the Tor network because each relay examine no more than one hop in the circuit neither an eavesdropper nor a negotiate relay can employ traffic analysis to link the connection's source and destination. Tor works for TCP streams and can be employed by any application with SOCKS support.

For efficiency, the Tor software employ the same circuit for communications that happen within the same ten minutes or so on. Later requests are given a new circuit, to carry people from linking our earlier actions to the new ones.

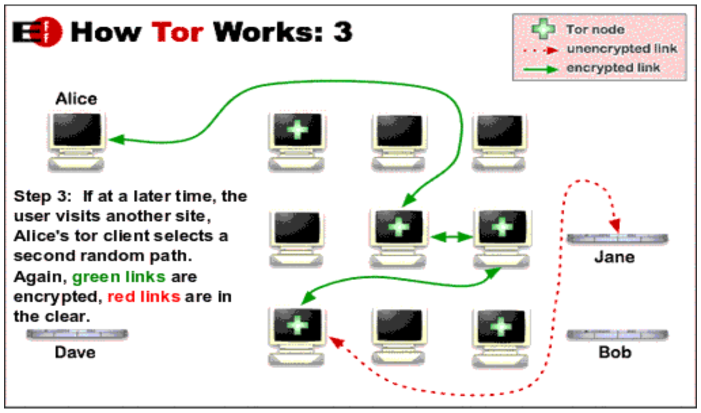
How Tor Works: 3
Tor node
unencrypted link
encrypted link
Alice
Step 3: If at a later time, the user visits another site, Alice's tor client selects a second random path. Again, green links are encrypted, red links are in the clear.
Jane
Dave
Bob

Figure 3: TOR WORKING

Tor cannot attempt to protect against monitoring of traffic at the boundaries of the Tor network (i.e., the traffic entering and exiting the network). While Tor does provide security against traffic analysis, it cannot assure traffic confirmation also called end-to-end correlation.

The Domain Name System (DNS) has become a critical and reliable component of the Internet, allowing individuals to rapidly match domain names with their corresponding IP addresses. The DNS is a hierarchical system,which at the top of the hierarchy is the root domain. Currently, the root consists of a combination of 13 groups of DNS servers located globally around the world. Each of those servers is named in the form [A-M].root-servers.net. These roots are responsible for the delegation of top-level-domains (TLDs) such as .com [7]. DNS leakage occurs when web traffic goes through the VPN or list of proxies, but address resolution is done locally by the ISP provider. Clearly, such address resolution outside the VPN can reveal attackers identify.

Attacker guides (see earlier section) specifically mention this with guidelines to make sure that DNS resolving is done through the VPN service. A DNS leakage often happens when there is a slight delay in the answer from the VPN DNS server or when the VPN fails to resolve the address.

This project explains a new cell-counting-based attack against Tor which allows the attacker to confirm anonymous communication relationship among users very quickly.Here the data is send from client to server system through onion router. This type of attack is highly efficient and can confirm very short communication sessions with only tens of cells. It is effective, and its detection rate approaches 100% with a very low false positive rate. It is possible to implement the attack in a way that appears to be very difficult for honest participants to detect.. Initially the client enters original IP address and then he is used Tor functionality change the IP address on that time he accounts will be blocked and the data will be safe. We can implement this project found the attackers and blocked the attackers at downloading the files.

Related work:

Murdoch and Watson [2] conducted an empirical analysis that investigated the relationship between the path selection algorithm and path compromise with respect to the attacks cost for the adversary. They identified the fraction of malicious Tor routers and the fraction of adversary-controlled bandwidth as important factors for predicting the adversaries ability to compromise paths. Browser based attacks on Tor was presented by Abbott et al. [3], the main idea is that the attacker can trick a users web browser into sending a peculiar signal over the Tor network and subsequently detected using traffic analysis. In the paper, they described how a malicious node acting as exit node, when selected by a client can insert a JavaScript code into unencrypted payload and transmit to the client, which then run on client machine and can generate identifiable signal pattern that can be detected by the server. Evans et al presented an attack based on legitimate guard node and malicious exit node [4]. In their attack guard node used by a legitimate client is kept busy through the long path generation. This push client to use malicious exit router to relays its traffic Bauer et al. [5] demonstrated how extend is the routing selection optimization for performance exposed Tor protocol to end-to-end traffic analysis attack from no global adversaries with minimum resources. Their approach involves compromising guard and exit nodes, by injecting few nodes with high bandwidth and high uptime claims, in a similar work, Bauer et al exploit the role of ports in compromising routing path based on the type of traffic being propagated [6].In this work we presented a new attack scenario that focus on Tor unpopular ports and combined inspiration from both Abbott et al. [3] and Bauer et al. [5]. Our work is similar to [5], in the sense that we both investigated the relationship of application layer protocol to the resources needed by adversary to compromise path, but differ in the definition of attack scenario; their attack scenario is passive one that involved injecting malicious entry and exit routers with high bandwidth claimed, while in our case is more of active attack to reveal the identity of a Tor client connected to a compromised web server; the attacker takes advantage of unpopular ports to play on the Tor path selection algorithm to select from few set of relays in which the attacker controlled a significant fractions a popular attack on Tor is to inject a certain number of malicious Tor nodes then to wait for a Tor client (victim) to pick one or several of these to construct his circuits. Currently the Tor network contains more than 3000 relays which makes those attacks not scalable anymore. In this paper we described a technique that allows to increase significantly the scalability of this type of attacks. The main idea is: instead of injecting typical Tor nodes (allowing typical port numbers e.g. 80, 53, etc.), we inject only Tor nodes allowing unpopular ports (e.g. 25, 119, etc.). Since only a small fraction of Tor nodes allow unpopular ports, the malicious injected Tor nodes will outnumber the valid ones which significantly increases the probability to compromise circuits created by Tor clients. The proposed attack is active in the sense that the attacker tries to push a typical Tor client to open new connections through unpopular ports. Empirical analysis based on data obtained from the real Tor network showed that by injecting 30 pairs of entry/exit relays and redirecting the traffic through unpopular ports, the anonymity of Tor clients is reduced by more than 50%. Given the few number of Tor nodes allowing unpopular ports, this attack is still scalable with the current Tor network (January 2013). The main requirement for this attack to work is a Tor client that uses an unpopular application (port) through the Tor network. Table I lists the unpopular ports in Tor. In the sequel we show how it is possible to actively push a typical Tor client (using typical ports) to open a connection through Tor using one of the unpopular ports[1]. Our attack differ from [3]; theirs is a browser based attack, that trick a users web browser to send a peculiar signals over Tor network, but our involve pushing Tor client to open a new connection through Tor network.

At Muhammad Aliyu Sulaiman and Sami Zhioua, Anonymity systems try to conceal the relationship between the communicating entities in network communication. Popular systems, such as Tor and JAP, achieve anonymity by forwarding the traffic through a sequence of relays. In particular, Tor protocol constructs a circuit of typically 3 relays such as no single relay knows both the source and destination of the traffic[1]. A known attack on Tor consists in injecting a set of compromised relays and wait until a Tor client picks two of them as entry (first) and exit (last) relays. With the currently large number of relays, this attack is not scalable anymore. In this paper, the advantage of the presence of unpopular ports in Tor network to significantly increase the scalability of the attack: instead of injecting typical Tor relays (with the default exit policy), we inject only relays allowing unpopular ports. Since only a small fraction of Tor relays allow unpopular ports, the compromised relays will outnumber the valid ones. at Muhammad Aliyu Sulaiman and Sami Zhioua show in paper [1] how Tor traffic can be redirected through unpopular ports. The experimental analysis shows that by injecting a relatively small number of compromised relays (30 pairs of relays) allowing a certain unpopular port, more than 50% of constructed circuits are compromised.

At Zhen Ling, Junzhou Luo,[8],Various low-latency anonymous communication systems such as Tor and

Anonymizer have been designed to provide anonymity service for users. In order to hide the communication of users, most of the anonymity systems pack the application data into equal-sized cells (e.g., 512 B for Tor, a known real-world, circuit- based, low-latency anonymous communication network). Via extensive experiments on Tor, we found that the size of IP packets in the Tor network can be very dynamic because a cell is an application concept and the IP layer may repack cells. Based on this finding, we investigate a new cell-counting-based attack against Tor, which allows the attacker to confirm anonymous communication relationship among users very quickly. In this attack, by marginally varying the number of cells in the target traffic at the malicious exit onion router, the attacker can embed a secret signal into the variation of cell counter of the target traffic. The embedded signal will be carried along with the target traffic and arrive at the malicious entry onion router. Then, an accomplice of the attacker at themalicious entry onion router will detect the embedded signal based on the received cells and confirm the communication relationship among users. We have implemented this attack against Tor, and our experimental data validate its feasibility and effectiveness. There are several unique features of this attack. First, this attack is highly efficient and can confirm very short communication sessions with only tens of cells. Second, this attack is effective, and its detection rate approaches 100% with a very low false positive rate. Third, it is possible to implement the attack in a way that appears to be very difficult for honest participants to detect (e.g.,using our hopping-based signal embedding). This attack is difficult to detect and is able to quickly and accurately confirm the anonymous communication relationship among users on Tor. An attacker at the malicious exit onion router slightly manipulates the transmission of cells from a target TCP stream and embeds a secret signal (a series of binary bits) into the cell counter variation of the TCP stream. An accomplice of the attacker at the entry onion router recognizes the embedded signal using our developed recovery algorithms and links the communication relationship among users. Our theoretical analysis shows that the detection rate is a monotonously increasing function with respect to the delay interval and is a monotonously decreasing function of the variance of one way transmission delay along a circuit. Via extensive real-world experiments on Tor, the effectiveness and feasibility of the attack is validated. Our data showed that this attack could drastically and quickly degrade the anonymity service that Tor provides. Due to Tor's fundamental design, defending against this attack remains a very challenging task that we will investigate in our future research.

Problem formulation:

In last paper, the Traffics going over Tor do not encrypted, it's just anonymous and also used malicious exit node and port or router can observe traffic and identified attacker/hacker. So, in this case, number of ports/routers is injected by attacker. This process takes more time to identify attacker

Many Tor users don't realize that when they use Tor, there is a many hazard that although connections across Tor network are torrified and encrypted, DNS requests made within the torrified application will not be routed via the Tor network. Therefore, the DNS requests will be built via our own machine. This is a security hazard because anyone watching our computer or our DNS server can tell what names we are looking up, and guess that those are the domains we are visiting via Tor. So, DNS requests not sent through Tor network by default and Attacker could see what websites are being visited.

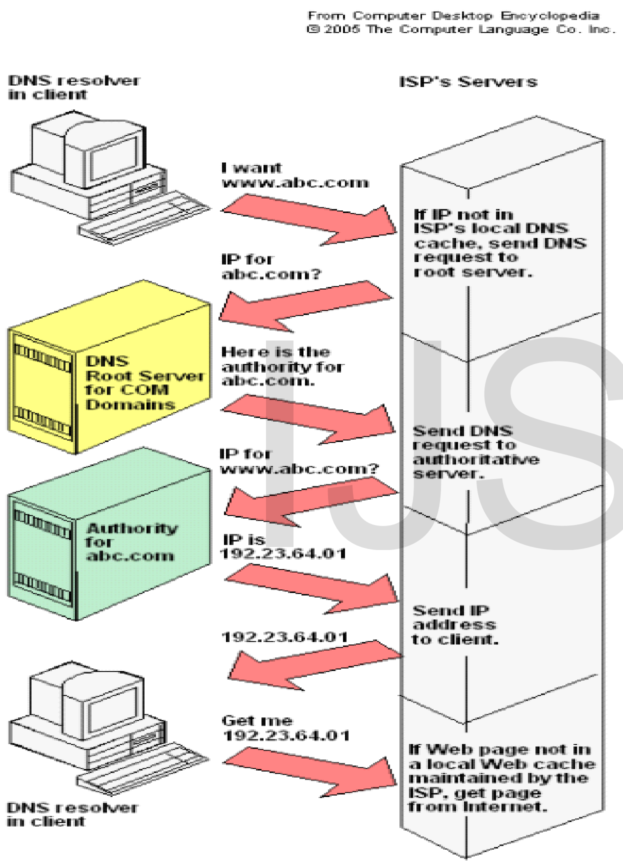
From Computer Desktop Encyclopedia
© 2005 The Computer Language Co. Inc.
DNS resolver in client
ISP's Servers
I want www.abc.com
If IP not in ISP's local DNS cache, send DNS request to root server.
IP for abc.com?
DNS Root Server for COM Domains
Here is the authority for abc.com.
Send DNS request to authoritative server.
IP for www.abc.com?
Authority for abc.com
IP is 192.23.64.01
Send IP address to client.
192.23.64.01
Get me 192.23.64.01
If Web page not in a local Web cache maintained by the ISP, get page from Internet.
DNS resolver in client

Fig5: DNS leaks in TOR

In the existing system the client can easily change his own IP address by using TOR functionality. The client gives his IP address then the TOR gives another IP address as the output which is incorrect. When his IP address is modified, he crashes the file of other. He cannot be tracked by the investigator of his crime because the IP address produced by TOR is nowhere available to any computer. To overcome all these problem to small changes of existing algorithm to proposed new technique.

Proposed work

We proposed Torsocks technique to rectify DNS leak in TOR network. No doubt, it is the best way to ensure that all of an application's network traffic passes through Tor. Torsocks can prohibit DNS leakage due to things like DNS prefetching or binary plugins that generate their own network traffic, whereas HTTP or SOCKS proxies cannot prohibit these kinds of leaks. On the other hand, proxies related Privoxy can be configured to send only certain traffic via Tor, whereas Torsocks forces all traffic through Tor.

If our application supports HTTP proxies, we might consider using Privoxy. Privoxy concentrating on local host port 8118, and the versions carried with Tor come preconfigured to forward traffic via Tor using SOCKS 4a. This is good, because employing SOCKS 4a causes DNS requests to be made remotely, and therefore does not leak DNS. So if we set it up properly, Privoxy itself does not leak DNS at all.

The admin tracks the IP address of the client, when the client is giving his own IP address to TOR functionality. The admin tracks the IP address before the TOR and compares it with IP address before downloading the file. When both the IP addresses are same then the admin gives all the access permissions for downloading the file else he blocks downloading of the file.

Conclusion and Future planning:

From our analysis we can say that all tested plugging can be used to disclose the real IP address of the user. Finally we conclude that this technique can reduce the attack in lasting time. This attack is difficult to detect and is able to quickly and accurately confirm the anonymous communication relationship among users on Tor. Due to Tor's fundamental design, defending against this attack remains a very challenging task that we will investigate in our future research. We found that increased traffic spikes within the global DNS for .onion requests corresponded with external global events, emphasizing the potential human factor in those leakages (i.e., user error). While the root cause of these leaked DNS queries remains unknown, our preliminary explorarion unveiled concerns to the severity of the leakage and to the possibility of more sensitive private information being unintentionally disclosed. Our future work will continue the examination of leaked DNS queries to the root but will also extend to other non-delegated TLDs such as i2p and .exit. We will plan to further dissect the impact of global events and the role of malware in the leakage, and investigate the potential privacy consequences of the leakage under the various leakage causes. By sharing this introductory work, we wish to trigger further discussion in the community.

Reference:

1. Muhammad Aliyu Sulaiman and Sami Zhioua, "Attacking Tor through Unpopular Ports," 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops
2. S. J. Murdoch and R. N. Watson, "Metrics for security and performance in low-latency anonymity systems," in Proceedings of the 8th international symposium on Privacy Enhancing Technologies, ser. PETS '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 115–132.
3. T. G. Abbott, K. J. Lai, M. R. Lieberman, and E. C. Price, "Browser-based attacks on tor," in Proceedings of the 7th international conference on Privacy enhancing technologies, ser. PET'07. Berlin, Heidelberg: Springer-Verlag, 2007, pp.184–199.
4. N. Evans, R. Dingledine, and C. Grothoff, "A practical congestion attack on tor using long paths," in Proceedings of the 18th USENIX Security Symposium, August 2009.
5. K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-resource routing attacks against Tor," in Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2007), Washington, DC, USA, October 2007.
6. K. Bauer, D. Grunwald, and D. Sicker, "Predicting tor path compromise by exit port," in Proceedings of the 2nd IEEE International Workshop on Information and Data Assurance (WIDA), Phoenix, USA, December 2009.
7. Mockapetris, P., Dunlap, K.J.: Development of the domain name system. Volume 18. ACM (1988)
8. Zhen Ling, Junzhou Luo, Member, IEEE, Wei Yu, Xinwen Fu, Dong Xuan, and Weijia Jia, "A New Cell-Counting-Based Attack Against Tor" IEEE 2012 Transactions on Networking, Volume:PP,Issue:99