# 1. Paper Title

The title of the paper is Tor: The Second-Generation Onion Router.

# 2. Authors

The authors of the paper are Roger Dingledine, Nick Mathewson and Paul Syverson. Roger Dingledine is in a leadership role with the Tor Project, as a project Leader, Director, and Research Director. Paul Syverson is an NRL researcher on most things. Nick Mathewson does a lot of the ongoing design work, and coordinates and leads ongoing development[2].

# 3. Date

The publication date of the paper was on 18 May 2004[3].

# 4. Novel Idea

The novel idea of the paper is to present Tor, a circuit-based low-latency anonymous communication service and the second-generation Onion Routing system with many improvements. The improvements are based on the addressed limitations in the original design.

# 5. Main Result(s)

The main results of the paper are the updated designs of original Onion Router design which are adding perfect forward secrecy, congestion control, directory servers, integrity checking, configurable exit policies and a practical design for location-hidden services via rendezvous points[1]. Furthermore, the paper proves the feasibility of the construction of the Tor system by discussing design details, attack and defense related issues and experiences: design goals and assumptions, the Tor design, rendezvous points and hidden services, other design decisions, attack and defenses, and early experiences: Tor in the wild[1].

# 6. Impact

The importance of these results is the inspiration of anonymous communication service development. It has great impacts for both theoretical and practical computer systems.

For theory of computer systems, the design details in the paper, especially those updated designs, allow developers to solve the problems and limitations existing in the old anonymous communication service, or create new applications. Those projects and ideas may be paused or abandoned before. However, the publication of this paper makes “a circuit-based low-latency anonymous communication service”[1] this theory workable again.

For practice of computer systems, these results present this is an extremely safe anonymous communication. Therefore, in practical computer systems, users are able to proceed anonymous communication without any risk such as being traced, identified, exploited or message disclosed.

# 7. Evidence

First of all, the authors use analytical analysis to justify the feasibility of the system by analyzing the goals of the design. For deployability, the design is not expensive to run in order to be deployed and used in the real world. For usability, the system is easy-to-use and with more anonymity in order to have more users. For flexibility, the protocol is flexible and well-specified so it can serve as a test-bed for future research. For simple design, the protocol’s design and security parameters are well-understood[1].

Secondly, the authors use both analytical and empirical analysis to address the non-goals. For not peer-to-peer, **Tarzan** and **MorphMix** are empirical analysis to address many short-lived servers that may be controlled by an adversary. For not secure against end-to-end attacks, Tor doesn’t claim to completely solve the timing or intersection attacks which is analytical analysis. For no protocol normalization, **Privoxy** and **Anonymizer** are empirical analysis to address that Tor does not provide protocol normalization like them[1].

Thirdly, the authors use analytical analysis in the discussion of the threat model which Tor doesn’t protect against strong adversaries.

Fourthly, the authors use different strategies to justify the results of the Tor design. For cells, the authors use a figure to demonstrate the structure of the cells. For circuits and streams, the authors use a figure to make a demonstration about how Alice builds a two-hop circuit and begins fetching a web page, in order to present the structure of the whole procedure. More specifically, the authors use mathematical formulas and procedures to illustrate constructing a circuit, and use analytical analysis to describe relay cells. For opening and closing streams, integrity checking on streams and rate limiting and fairness, the authors use analytical analysis to establish the result by presenting the procedures and structures of the units. For congestion control, the authors use analytical analysis which is parsing both circuit-level throttling and stream-level throttling to justify the results.

Fifthly, for rendezvous points and hidden services, the authors use empirical analysis which is listing the steps in the procedure to justify the result rendezvous points in Tor, and use analytical analysis to establish the results in integration with user applications and previous rendezvous work sections by illustrating the workflow.

Sixthly, in the other design decisions section, the authors use analytical analysis for denial of service, exit policies and abuse and directory servers to establish the results which are the possible attack methods.

Seventhly, for attacks and defenses, the authors use analytical analysis for passive attacks, active attacks and directory attacks by listing the potential breaches under three different attack methods.

At last, the authors use empirical analysis, especially many data such as dates, percentages and numbers, to establish the results about early experiences: Tor in the Wild.

# 8. Prior Work

This paper’s results are based on several related works.

The first work is the Onion Routing project which was published before, several design and analysis papers are included but many critical design and deployment issues were never resolved[1], and not been updated in years. The original design of Onion Routing is a distributed overlay network, designed to anonymize TCP-based applications.The clients choose a path through the network and build a circuit which each node in the path knows its predecessor and successor, but the other nodes in the circuit does not. The traffic flows down the circuit in fixed-sized cells and unwrapped by a symmetric key at each node and relayed downstream[1].

To be more specific, the modern anonymity systems are based on Chaum’s **Mix-Net** design. By wrapping messages in layers of public-key cryptography and relaying them through a path composed of mixes, the correspondence between sender and recipient can be hidden[1]. There are several system examples mentioned in the paper that can be seen as the source of inspiration of the Tor design: **Babel**, **Mix-master** and **Maxminion**[1].The disadvantage of these systems are high-latency.

However, Tor’s design belongs to low-latency designs that try to anonymize interactive network traffic. By handling a variety of bidirectional protocols[1]. There are two types of low-latency designs: single-hop proxies such as the **Anonymizer**, and distributed-trust, circuit-based anonymizing systems. A lot of systems are mentioned in the paper, including both single-hop proxies and distributed-trust, circuit-based anonymizing systems: **Java Anon Proxy**, **PipeNet**, **ISDN mixes**, **Tarzan**, **MorphMix**, **Crowds**, **Hordes**, **Herbivore**, **P5**, **Freedom**, **Cebolla** and **Anonymity Network**[1]. They can be considered as the foundation of the Tor’s design.

# 9. Competitive work

There are 9 major improvements in Tor: The Second-Generation Onion Router compared to related prior work.

The first one is separation of “protocol cleaning” from anonymity. In the original design of Onion Routing, it requires a separate “application proxy” for each supported application protocol which was never supported. So Tor uses the standard and near-ubiquitous SOCKS proxy interface which allows them to support most TCP-based programs without modification[1].

The second one is No mixing, padding, or traffic shaping. In the original design of Onion Routing, it called for batching and reordering the cells as they arrived and added padding between onion users and ORs later[1]. Thus, Tor abandoned these strategies due to the impractical and uneconomical facts of these resources being used.

The third improvement is Many TCP streams can share one circuit. In the original design of Onion Routing, it built a separate circuit for each application level request so it required multiple public key operations for every request which threatened anonymity. So Tor multiplexes multiple TCP streams along each circuit to improve efficiency and anonymity[1].

The fourth improvement is Leaky-pipe circuit topology. Tor initiators can direct traffic to nodes partway down the circuit, which allows traffic to exit the circuit from the middle[1], which is not a part in the original design of Onion Routing.

The fifth one is Congestion control. In earlier anonymity designs, it doesn't address traffic bottlenecks. Tor decentralized congestion control by using end-to-end acks to maintain anonymity[1].

The sixth one is Directory servers. In the earlier design of Onion Routing, it planned to flood state information through the network which can be unreliable and complex. Tor makes certain more trusted nodes act as directory servers which provide signed directories and allow users to periodically download the known routers and their current state via HTTP[1].

The seventh improvement is Variable exit policies. Tor provides a consistent mechanism for each node to advertise a policy describing the hosts and ports to which it will connect[1], which is not a part in the original design of Onion Routing.

The eighth improvement is End-to-end integrity checking. In the original design of Onion Routing, there is no integrity checking on data. Tor verify data integrity before it leaves the network[1] to prevent attacks.

The last one is Rendezvous points and hidden services. In the previous design of Onion Routing, it included long-lived “reply onions” that could be used to build circuits to a hidden server but didn’t provide forward security. Tor provides an integrated mechanism for responder anonymity via location-protected servers by negotiating rendezvous points to connect with hidden servers[1].

# 10. Reproducibility

Yes, the finding can be reproduced.

In the Tor design section, for cells, there is a figure demonstrating the structure of the cells. For circuits and streams, there is a figure demonstrating the workflow of a sample two-hop circuit and begins fetching a web page. For constructing a circuit and relay cells, the authors use step by step descriptions to guide the construction, such as *“To begin creating a new circuit, the OP (call her Alice) sends a create cell to the first node in her chosen path (call him Bob)...To extend the circuit further, Alice sends a relay extend cell to Bob, specifying the address of the next OR (call her Carol), and an encrypted gx2 for her...”*[1]. For opening and closing streams, integrity checking on streams, rate limiting and fairness and congestion control, the authors use the same technique such as “The OP then opens the stream by sending a relay begin cell to the exit node, using a new random streamID. Once the exit node connects to the remote host, it responds with a relay connected cell. Upon receipt, the OP sends a SOCKS reply to notify the application of its success.”[1]. Especially in congestion control, the authors use *“To control a circuit’s bandwidth usage, each OR keeps track of two windows. The packaging window tracks how many relay data cells the OR is allowed to package...”*[1] for circuit-level throttling and *“ORs and OPs use relay sendme cells to implement end-to-end flow control for individual streams across circuits. Each stream begins with a packaging window (currently 500 cells), and increments the window by a fixed value (50) upon receiving a relay sendme cell...”*[1] for stream-level throttling.

In rendezvous points and hidden services, the authors list a step by step procedure: *”Bob generates a long-term public key pair to identify his service. “Bob chooses some introduction points, and advertises them on the lookup service, signing the advertisement with his public key. He can add more later...”*[1] for rendezvous points in Tor. For integration with user applications and previous rendezvous work, the authors use similar techniques to present the procedure.

In other design decisions section, the authors describe detailed procedures of all three possible attack methods: denial of service, exit policies and abuse and directory servers. Such as *“...Exit abuse is a serious barrier to wide-scale Tor deployment. Anonymity presents would-be vandals and abusers with an opportunity to hide the origins of their activities…”*[1].

For attacks and defenses, the authors list all parts which can be potential breaches of the system for all three attack modes: passive attacks, active attacks and directory attacks. For example: *“...Observing user content...Option distinguishability...End-to-end timing correlation”*[1].

For early experiences: Tor in the wild, the authors use many numbers such as *“As of mid-May 2004, the Tor network consists of 32 nodes (24 in the US, 8 in Europe)... the current remailer network has about 40 nodes.) Each node has at least a 768Kb/768Kb connection, and many have 10Mb...Each Tor node currently processes roughly 800,000 relay cells (a bit under half a gigabyte) per week. On average, about 80% of each 498-byte payload is full for cells going back to the client, whereas about 40%...”*[1] to prove the result.

In conclusion, these results have enough detailed information that can support reproducing the results.

# 11. Question

Is Tor wasting our precious electricity resources?

According to the design of Tor, it is better if the Tor network has more nodes and nodes must be online in order to transmit data. However, data transmission and keep nodes online both cost power, in other words, electricity. There are on average 7,000 relays online all the time[4], and each computer costs about $100 worth electricity per year[5]. So each year there is at least $700,000 electricity wasted in maintaining the Tor network. It is called waste because the data could be directly transmitted from end-to-end instead of going through many relays. To make the anonymous communication come true, this is an expensive cost of power supply.

# 12. Criticism

Tor can be used in criminal or terror activities.

The primary expectation of the invention of Tor is to create a safe, unsupervised way to exchange data and visit the internet. However, Tor is a double-sided sword, it not only protects normal users’ privacy, but also helps hiding criminals and terrorists’ activities. According to Andrew Lewman, as executive director of Tor: *”What’s changed most about Tor is the drug markets have taken over…We had all these hopeful things in the beginning but ever since Silk Road has proven you can do it, the criminal use of Tor has become overwhelming. I think 95 percent of what we see on the onion sites and other darknet sites is just criminal activity. It varies in severity from copyright piracy to drug markets to horrendous trafficking of humans and exploitation of women and children.”*[6]. Furthermore, the existence of Tor also threaten the national security: *“While Tor and Bitcoin have extremely legitimate uses for many users especially those in repressive regimes, both systems are often referred to in relation to cybercrime...Systems available to us are also available to those that oppose us and cryptocurrencies such as Bitcoin are reportedly being used by the likes of ISIS.”*[7]. Therefore, the existence of Tor is controversial.

# 13. Ideas for further work

Instead of making Tor public so it may threaten the nation and society's security, my idea is to use Tor with an invitation-only method in a closed environment such as inside a company, school, game community, etc.

As long as each new node needs to be invited to join the network, the Tor system is under control. Although no one can identify the others, the whole community is safe because only related users can receive the invitation. The connection between nodes is outside of the network, which makes the existence of the system based on a reason and all nodes are related to it. When nodes send messages, they don’t have to worry about being identified due to the Tor’s design: their messages are encrypted and go through other nodes’ relays. So they can say something they are afraid or inconvenient to say with the original identity. For example, point out someone’s mistake, raise a risky suggestion, chat with virtual friends in game, etc.

# 14. References

1. Roger Dingledine, Nick Mathewson, Paul Syverson. “Tor: The Second-Generation Onion Router”. 18 May 2004

2. Torproject. “About People”. 5/30/2020. https://www.torproject.org/about/people/

3. Usenix. “Security '04 Paper”. 18 May 2004

4. Metrics. “Servers”. 5/30/2020. https://metrics.torproject.org/networksize.html

5. Vivintsolar. “How Much Electricity Does a Computer Use”. 11/28/1. https://www.vivintsolar.com/blog/how-much-electricity-does-a-computer-use

6. Liquidvpn. “Tor's ex-director: 'The criminal use of Tor has become overwhelming'”. May 12, 2015. https://www.liquidvpn.com/isis-use-tor-and-bitcoin-to-fund-terrorism/

7. Patrick Howell O'Neill. “ISIS use Tor and Bitcoin to fund terrorism”. MAY 22, 2017. https://www.cyberscoop.com/tor-dark-web-andrew-lewman-securedrop/