Summary of the paper Tor: Second-Generation Router

anonymous authors

Hints from etherpad

0.1 Reading

- read your paper multiple times
- find out when and where it has been published
- understand the relevant basics of the field
- search for other papers on the same topic to judge the relevance and contribution of your paper
- published at an earlier time or at a later time
- dealing with other problems in the same field
- dealing with similar problems in different fields
- tackling a similar challenge or other challenges

0.2 Writing

- Write up a summary with your findings
- length restriction: fit it into 4 pages A4 max!
- two-column 11pt Times!
- please use LaTeX for the layout (see template below)
- You may (re-)use figures or formulas whenever appropriate (if referenced correctly)
- Do not re-use text from the paper (no copy&paste)!
- upload your summary by the deadline to the skiconf tool
- authors remain anonymous => do NOT include your names!
- bonus: ensure that the PDF metadata does not disclose your login name etc.
- language: English

0.3 General hints for summary

- Briefly outline the "field" this paper belongs to, the challenges that exist in this field and how your paper fits into the bigger picture
- Explain the relevant basics and fundamentals
- What stuff do you have to know in order to understand the content of the paper and to judge its impact?
- Your summary should contain answers to the following:
- What are the problems to be solved? Why are they worthwhile to solve?
- What methods are used? How is the problem tackled?
- What results are obtained? What do the results mean?
- Additionally: outline the impact of the paper
- What is the "delta" that the authors of the paper achieved in comparison of the state of the art at the time the paper was published
- How is it different/better compared to related works?
- How has the paper been received by the scientific community?
- How do papers that have been published later talk about this paper?
- Is the delta of the paper of relevance today?
- Did the paper influence its field?

0.4 General advice

- be as descriptive and concrete as necessary, but still as concise as possible
- give examples if suitable
- most important challenge
- your Summary should be *understandable on its own*
- you will have to leave out lots of (less relevant) details
- but: you may have to add things that are not described in the paper
- ask yourself: does the summary contain the most relevant contributions of the paper?
- do not confuse your summary with the "Abstract" at the beginning of a paper
- your summary contains more details than the abstract of the paper
- your summary contains information *about* the paper (which was not available when the paper was published)
- remember: do not copy text word by word from your paper (plagiarism)!!!

1 Introduction

The paper *Tor: Second-Generation Onion Router* was written by Roger Dingledine, Nick Mathewson and Paul Syverson in the year 2004. It describes a circuit based low-latency anonymous communication system called Tor. Tor based on the concept of Onion Routing, which was first published in the paper *Hiding routing information* in 1996 [1]. Onion Routing is also a low latency and circuit based communication system. A low latency network is a network that experiences small delay times [2]. Because of that Tor is able to anonymize interactive network traffic and can used in the world wide web. Circuit based means that Tor establishes circuits across the network to communicate.

2 Onion Routing/Technical background

In [3] David Chaum presented the mix concept which used the at that time freshly invented priciple of public key cryptography to build the theoretical foundation of untraceable electronic communication.

In 1995 the work on developing onion routing began - a technique to offer anonymous connections[1] using some of the concepts of Chaum [4, 1, 5]. In the onion routing network an initiating application makes connections through a sequence of machines called onion routers. The data sent by the application consists of layers of encrypted data (like an onion) containing the next destination of the message in the network. Each layer of the onion is decrypted by one onion router and contains the next hop in the route. So each onion router knows only its adjacent onion routers. Because of that an onion router is not able to know who is communication with whom.

Then in 2004 the original Tor paper was published [6] and the Tor network was deployed.

3 Tor

The details mentioned in this section are taken from [6] if not stated otherwise.

Tor is a circuit-based low-latency service providing anonymous communication improving the concepts of the already explained onion routing. It is designed to anonymize TCP connections. This enables Tor to work with most standard applications of the net like HTTP to visit websites and other application layer protocols.

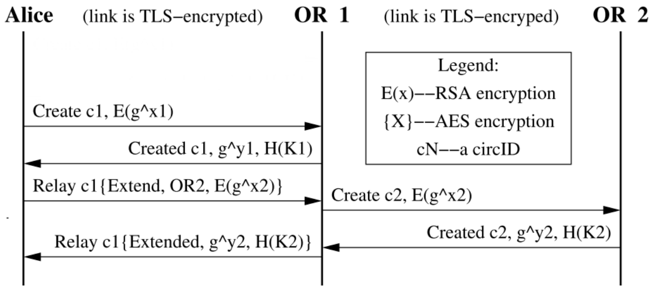

Figure 1: Creating a circuit with two onion routers. Taken from [6] and slightly edited.

Users of the network use an **onion proxy** to connect their TCP connections to the Tor network. They establish a circuit through the network containing several nodes, so-called **onion routers**. Every onion router just knows his predecessor and successor in the circuit. How this works in detail is covered below. An overview about the exchanged messages can be found in figure 1.

3.1 Creating circuits

Tor uses TLS connections between the onion routers to prevent message tampering by external adversaries and providing perfect forward secrecy (see next section). A client who wants to use the network has to establish a circuit consisting of several onion routers incrementally. Let's call the client Alice and the first onion router on her chosen circuit Bob[2]. Bob (as well as every other onion router) owns a long-term identity key pair to sign TLS certificates and his router description and a short-term onion key pair used when setting up circuits. .

Alice sends Bob a *create* control cell[3] containing a unused circuit id and a DH[4] key exchange parameter g^x encrypted with Bob's onion key. Bob decrypts this parameter, computes the shared key $K = g^{xy}$ and sends Alice a *created* cell containing the circuit identifier and the DH parameter g^y so that Alice can compute K, too.

When Alice wants to extend this circuit to the next onion router Carol she sends a relay extend cell to Bob, which contains the address of Carol as well as a DH parameter g^{x_2} encrypted with Carols onion key. This data is encrypted symmetrically with AES-128 and the already negotiated key K between Alice and Bob.

Bob decrypts this information, chooses a unused circuit

[1] The communication using these connections does not have to be anonymous. The goal is to limit the traffic analysis in a way that makes it impossible to determine who is communicating with whom through a public network.

[2] Traditions have to be preserved!

[3] Tor cells have a fixed size of 512 bytes and always contain circuit identifier, the command and a payload. The command can be *control* for cells which are interpreted by the receiving onion router or *relay* for carrying end-to-end data.

[4] The Diffie-Hellman key exchange is a method to exchange keys over a public network using public-key cryptography. Details can be found in [7].

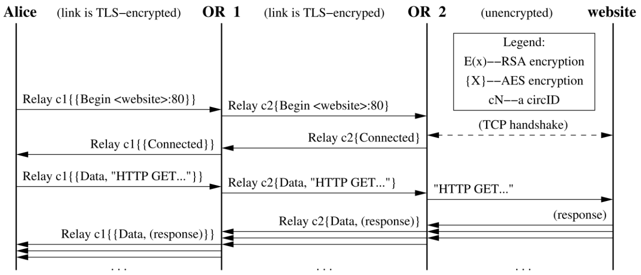

Figure 2: Requesting the content if a website via HTTP over an established circuit. Taken from [6] and slightly edited.

identifier between Carol and him and forwards the information received from Alice to Carol in a new *create* cell. The same process as already described takes place on Carols side and she sends g^{y_2} back to Bob after computing the key $K_2 = g^{x_2 y_2}$. Bob encrypts this information with K and sends it back to Alice who can also compute K_2 then. The circuit is extended by another onion router now and Alice has a key shared with Bob and one shared with Carol.

3.2 Sending TCP data through a circuit

After having established a circuit Alice can start to send data over the Tor network. This is illustrated below for the simple case of requesting a website.

4 Improvements to onion routing

In case of Tor the usability is a security requirement because it hide users among users and a system with fewer users provides less anonymity.

Additionally to onion routing, Tor added perfect forward secrecy, congestion control, directory servers, integrity checking, configurable exit policies and a practical design for location-hidden services via rendezvous points.

Perfect forward secrecy: In the original onion routing design a single hostile node can record traffic and later compromise succesive nodes in the circuit. To avoid this Tor uses an incremental path building design. Thant means, that the initiator negotiates session keys with each succecive node in the circuit. If these keys are deleted, the compromised nodes are not able to decrypt old traffic.

Congestion control: If many users choose the same OR-to-OR connectionin their circuits, that connection can be saturated. Without any congestion control this bottleneck can propagate through the whole network. To avoid this, Tor is able to control the circuits bandwidth in each OR.

Directory servers: Both Onion routing and Tor has to send periodically state informations (known ORs with their current states) to their users. In the onion routing design they flood their network with these state informations. Tor uses so called *directory servers* to distribute this informations through the network. The users download them periodically every 15 minutes via HTTP.

Integrity checking: Tor uses TLS between its nodes. Because of that an adversary is not able to change the content of the data.

Rendezvous points: If a user wants to connect with a hidden server the onion routing design uses long lived "reply onions", which are used to build a circuit to a hidden server. This is bad because the onion routing design does not provide forward security. For a connection between an user and a hidden server, tor uses so called rendezvouz points. If a user wants to create a connection to a hidden server, it uses the introduction points of this hidden server to send them a rendezvous point. The user and the hidden server uses this rendezvous point to create a circuit between them. Every hidden server has circuits to ORs, which are introduction points.[8]

5 Importance/Impact

This paper has a big impact on the computer since and was cited more than 3000 times by other papers. A wide area onion routing network was briefly deployed but it was not really used because it required a seperate "application proxy" for each supported application protocol and most of them where never written. Tor uses the standard and near-ubquitous SOCKS proxy interface, which support most TCP-based programs without modification. Because of that tor is used in the world wide web since 2004 for anonymous connections. Individuals use Tor for censorship circumvention to reach othervise blocked content, to keep websites from tracking them or their family members, to publish services without needed to reveal the location of them or for sensitive communication. Tor is also used by companies or Journalists to communicate with whistleblowers[9]. But Tor is also used by criminals to sell drugs, child porns, weapons or other illegal things.

References

[1] D. M. Goldschlag, M. G. Reed, and P. F. Syverson, “Hiding routing information,” in *International Workshop on Information Hiding*, pp. 137–150, Springer, 1996.

[2] “Introduction to latency on computer networks.” `https://www.lifewire.com/`

latency-on-computer-networks-818119. Accessed: 2016-11-17.

[3] D. L. Chaum, “Untraceable electronic mail, return addresses, and digital pseudonyms,” *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

[4] “Onion routing.” https://www.onion-router.net/Summary.html. Accessed: 2016-11-20.

[5] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, “Anonymous connections and onion routing,” *IEEE Journal on Selected areas in Communications*, vol. 16, no. 4, pp. 482–494, 1998.

[6] R. Dingledine, N. Mathewson, and P. Syverson, “Tor: The second-generation onion router,” in *Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM’04, (Berkeley, CA, USA), pp. 21–21, USENIX Association, 2004.

[7] W. Diffie and M. Hellman, “New directions in cryptography,” *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[8] “Tor: Hidden service protocol.” https://www.torproject.org/docs/hidden-services.html.en. Accessed: 2016-11-17.

[9] “Tor: Overview.” https://www.torproject.org/about/overview.html.en. Accessed: 2016-11-17.