The Tor Network - Design and Defenses

Computer Security 1 - CS5460

LaTeX

Jesse Victors

November 2013

I. Introduction

The Tor network is a second-generation onion routing system that aims to provide anonymity, privacy, and Internet censorship protection to its users. Tor, an open-source project run by a non-profit organization, routes encrypted TCP traffic through a worldwide network of over four thousand relays run by volunteers across the world. Tor's encryption, authentication, and routing protocols are designed to make it infeasible for any adversary to identify an end user or reveal their traffic. Throughout its history, various organizations and governments have attempted to block, tap, or crack the Tor network. It was recently revealed that the US National Security Agency has attempted to penetrate the Tor network. Tor is also currently undergoing a transition from RSA-based to elliptic-curve-based TLS. In light of these attacks, Tor's popularity, and the protocol transition, a question that must be asked both by its users and by outsiders is: how secure is Tor? How does it work, what does it provide, and what are its weaknesses? In this paper, I attempt to address these questions.

II. Design

Tor provides an anonymity and privacy layer by relaying all end-user TCP traffic through a series of *relays* on the Tor network. Typically this route consists of a carefully-constructed three-hop path known as a *circuit*, which changes over time. These nodes in the circuit are referred to as *entry guard*, *middle router*, and the *exit node*, respectively. Only the first node can determine the origin of TCP traffic through Tor, and only the exit node can examine the contents and its destination. Nodes in the middle are unable to determine either. No single node can determine the origin, the contents, and the destination of traffic through the network. Tor's architecture is designed to make it exceptionally difficult for a well-resourced adversary to uncover the identity of the end-user and their network activities, even if nodes are compromised.[6]

A. Routing

In traditional Internet connections, the client communicates directly with the server. In this model, an eavesdropper can often reveal both the identity of the end user and their activities. Direct encrypted connections do not hide IP headers, which expose source and destination addresses and the size of the payload. In the face of adversaries with sophisticated traffic analysis tools, such information can be very revealing for someone who wishes to hide their activities altogether.

Tor combats this by routing end user traffic through a randomized path through the network of relays. To construct this circuit, the Tor client software first queries a trusted directory server, which return a list of Tor nodes.[8] This is illustrated in Figure 1.

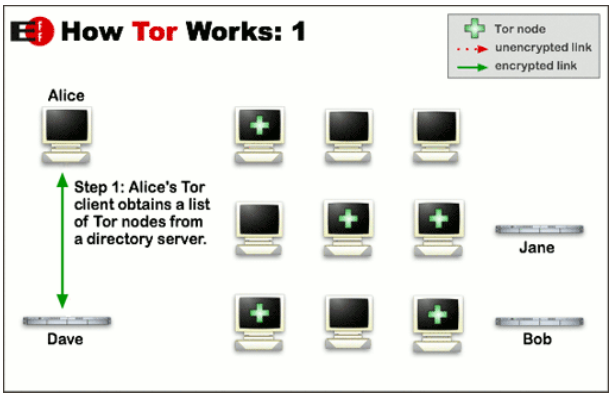

Fig. 1. The user's software first downloads a directory list of Tor relays. This information is later used to construct a circuit through the network.[6][18]

The second step involves choosing three nodes to use and carefully constructing an encrypted path between them. The circuit is extended one hop at a time such that no single relay ever knows the complete path. The client negotiates a separate set of encryption keys for each hop along the circuit to ensure that each hop cannot trace these connections as they pass through, as seen in Figure 2.

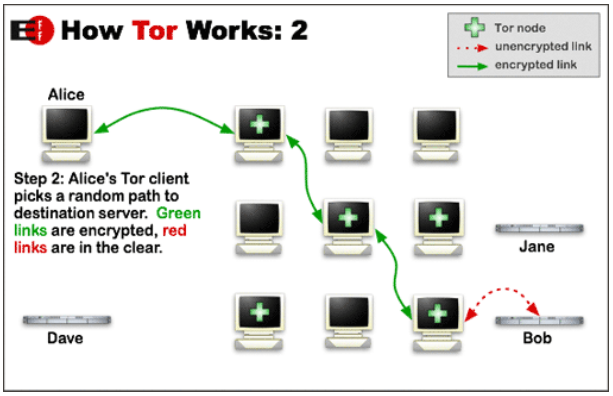

Fig. 2. A Tor circuit is incrementally constructed using layers of encryption. Each node has limited visibility, and no individual node knows the whole circuit.[18]

Following the complete establishment of a circuit, the Tor

client software then offers a Secure Sockets (SOCKS) interface which multiplexes TCP traffic through Tor. As each relay sees no more than one hop in the circuit, neither an eavesdropper nor a compromised relay can use traffic analysis to link the connection's source and destination. Tor further obfuscates user traffic by changing the circuit path every ten minutes,[6] as shown in Figure 3.

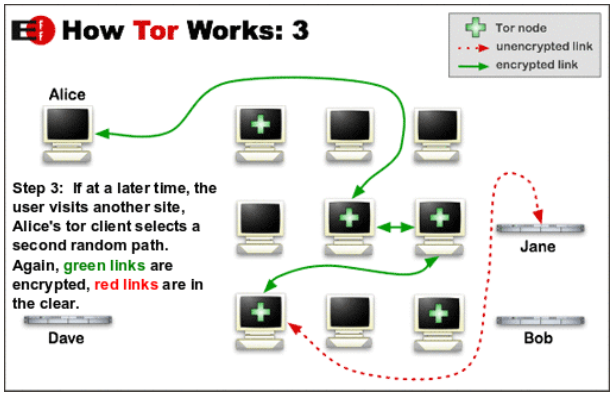

Fig. 3. A Tor circuit is changed periodically, essentially providing a new identity to the end user.[18]

It should be noted that traffic exiting the Tor network is often (but not always) encrypted on its way to the source. An outsider is therefore faced with up to four layers of TLS encryption. Currently, Tor is one of the most secure tools to use against network surveillance, traffic analysis, and to bypassing information censorship.

B. Encryption

Encryption is a necessary component for privacy within the Tor network. Traffic passing between nodes must be secured from outsiders, and even compromised nodes must not be able to observe the traffic in cleartext. Privacy is also important for traffic outside the network; encrypted connections between the exit node and the target web server must also be achieved. Furthermore, two layers are needed: one for inside the network and one for outside it.

Within the network, nodes talk to each other via the Transport Security Layer (TLS) protocol. When a circuit is built, each pair of nodes within it must first come to an agreement over an encryption key and the specific cipher to use for encryption. There are several mechanisms by which this can be done, all of which use Diffie-Hellman (DH) for key exchange. The most common methods include: TLS_RSA, which rely upon public and private keys generated with RSA; TLS_DHE, ephemeral DH; TLS_ECDH, DH based on elliptic curves, and TLS_ECDHE, ephemeral elliptic curve DH. It should be noted that only TLS_DHE and TLS_ECDHE provide perfect forward secrecy, so it is no surprise that Tor exclusively prefers them. Once encryption keys and the specific cipher (as well as its mode) have been agreed upon, communication of traffic between two nodes can be encrypted and exchanged.

Tor users typically use the Tor Browser Bundle, (TBB) a custom build of Mozilla Firefox with a focus on security and pricacy. The TBB not only provides special handling of client-side scripts such as Javascript, but also offers the HTTPS Everywhere extension, which uses regular expressions to rewrite web requests into ones that use HTTPS. Thus, if the web server is capable of handling SSL or TLS connections, HTTP communications will be encrypted under them. Like any typical browser, the TBB negotiates with the web server for the cipher and keys required for SSL/TSL, except that this negotiation entirely occurs through the Tor circuit. As previously noted, this information, along with subsequent HTTP data, is encrypted between each node in the circuit. During transmission, each node in the circuit decrypts its layer in turn, until the final node (the exit) passes the data to the target server. The TLS/IP connections remain open, so the returned information likewise travels back up the circuit to the end user.

In September 2013, Robert Graham of Errata Security analyzed 22,920 incoming connections to his exit node and found that 89.9% of the circuits agreed to use the Advanced Encryption Standard (AES) block cipher in cipher-block chaining (CBC) mode. This is the most common mode for AES, and allows for parallel decryption of the data blocks. The other 10.1% of the circuits relied upon three rounds of the Data Encryption Standard (triple DES) cipher for encryption. Furthermore, 75.7% of the circuits used elliptic-curve DH, with the remaining 22.3% relying upon traditional RSA DH.[13] Both protocols are discussed below.

1) RSA: RSA is an algorithm for public-key cryptography. Its security is based on the infeasibility of factoring the product of two large primes. RSA, like all other public-key cryptography algorithms, relies upon two keys: one public and the other private. The public key may be published and is used for encryption and for the verification of digital signatures. The private key is used for decryption and the generation of digital signatures. Thus, only the owner of the private key can decrypt incoming messages, and only they can digitally sign outbound messages. These properties make RSA and other public-key algorithms extremely powerful as they provide authenticity, integrity, and privacy of communications between two parties.

Tor version 2.3.x and previous rely upon TLS_DHE for the key exchange mechanism. Per the TLS specification, the server generates a Diffie-Hellman public key and sends it to the client. The server also uses its RSA private key to sign everything that it sends to the client during the DH exchange. The client can then confirm the digital signature to verify the authenticity of the communication from the Tor relay.

2) Elliptic-curve: Elliptic Curve Cryptography (ECC) is an approach to public-key cryptography in which elliptic curves are used instead of RSA. In contrast to RSA, ECC relies upon the infeasiblity of finding the discrete logarithm of a random elliptic curve element with respect to a publicly known base point. This is known as the elliptic curve discrete logarithm problem, or ECDLP.

A smaller key size is one of the most significant benefits introduced by ECC. Current NIST recommendations state that a key size of 160 bits for ECC offers a similar level of security to 1024 bits of RSA/DH. Likewise, 224 bits for ECC is analogous to 2048 bits of RSA/DH. It should be noted that the required key sizes for ECC increases significantly

slower than the key sizes for RSA and Diffie-Hellman. Thus ECC offers more security per increase in key size than for RSA. The smaller key size also directly correlates to an increase in computational efficiency; as the bit size increases, the speed difference between Diffie-Hellman and ECC grows superlinearly.[9]

Tor 2.4.x introduced TLS_ECDHE, which uses an elliptic-curve variant of the Diffie-Hellman key exchange. Like TLS_DHE, the server uses its RSA key to digitally signs communication to the client. The TLS_ECDHE protocol is believed to be more secure than TLS_DHE.

3) Symmetric-key: Once the SSL/TLS handshake completes, both parties use the generated session key and turn to symmetric-key encryption for hiding of data. Common symmetric-key algorithms include Data Encryption Standard, Advanced Encryption Standard, and RC4.

Both DES and AES are block ciphers, whereas RC4 is a stream cipher. DES, shown to be susceptible to brute-force attacks in 1998, has been replaced by triple-DES, (3DES) which itself has been succeeded by the much-stronger AES cipher. However, 3DES is still used on the web and by older Microsoft products.[4][10][7] The RC4 cipher is simple and extremely fast in software, and is now the most widely used software stream cipher, most notably in protocols such as TLS and WEP.

Robert Graham's analysis of the incoming handshakes to his exit node revealed that almost 90% of the circuits agreed on the AES cipher, while the remainder used 3DES. For communication to web servers, RC4 remained the most popular algorithm.[13]

III. Attacks and Defenses

Attacks from adversaries against Tor can be classified into several categories: attack on the user, attacks on their computer, attacks on the Tor network itself, attacks on the exit's connection across the Internet, and attacks on the server. Tor's developers have implemented or recommended various defenses against common attacks on each of these layers.

A. Social engineering

Human mistakes are one of the most serious threats to deanonymization, and defenses against it are outside the scope of the Tor project. While Tor users can enjoy a reasonable level of anonymity and protection against traffic analysis, use of the Tor network is useless if information about their identity and/or location is leaked to the Internet. The electronic trail they leave behind can be used to completely deanonymize them and reveal their activities. The Tor Project provides recommendations that aim to prevent users from leaking their identity, but actual defenses against this attack are generally outside the scope of Tor.

(notable examples, describe defenses)

B. Malware

Tor users can also be revealed or have their privacy compromised by their own workstations. If their computer contains spyware, backdoors, or other malicious software that can be used to identify them or their traffic, then a user may be tricked into believing that they are anonymous when in fact they are not.

(going to expand this a bit more, discuss Javascript vs NoScript, cross-site scripting attacks, TBB defenses)

C. Attacks against the Tor network

The relay chosen for the first hop in the circuit has knowledge of the user's IP address, can thus deanonymize them in that respect. Although this first relay does not know the rest of the circuit and cannot deduce the final traffic exiting Tor, the knowledge of the user's IP can be problematic if the relay is operated by an attacker. With this information, an adversary would be able to trace users or deny them entry into the Tor network.

Tor attempts to combat this through *entry guards*, a feature introduced in May 2006. Directory servers mark relays as guards as both being fast (i.e. has bandwidth above the median of all other relays) and stable (i.e. averages and uptime above the median of all other relays). The Tor client software chooses three guards out of the pool and then exclusively picks from these chosen relays as the first hop of any circuit.[8] The goal of this selection is to protect the end user against the "predecessor attack", wherein the attacker can preform end-to-end correlation and deanonymize the user if compromised nodes are chosen for the first and last hop in the circuit.[1] As relays initially are only the middle hop, this approach also makes it more difficult for an attacker to add relays to the Tor network and immediately begin tracing users. As of the time of this writing, there are 4,737 relays on the Tor network, which means that the chances of a client including a compromised node in its selection pool is 4,737 choose three without replacement, or 0.06334%.

To achieve its low-latency objective, Tor does not explicitly re-order or delay packets within the network.[6] If an adversary controls both the first hop and the final hop in the Tor circuit, then it may be possible for the attacker to perform timing attacks. To do this, the attacker would listen for incoming packets from a certain IP and simultaneously keep track of traffic flow out of the exit node. They could then use statistics to gain a reasonable correlation between the incoming and outgoing traffic. This would result in a deanonymization of the end-user and, if the connection to the web server was not encrypted, reveal their traffic. At its onset Tor was susceptible to this timing attack, but as the size and usage of the Tor network increased, the likelihood of successfully launching such a traffic analysis attack quickly became negligible.[8]

D. Wiretapping the exit node

Exit nodes are valuable on the Tor network due to their necessity for the final hop of the circuit and because they are rarer than traditional relays. Exit node operators typically have to deal with ISP complaints for any nefarious Internet activities performed by the users at the other end of the circuit. Although these complaints can be largely combated by implementing a reduced exit policy, there are 912 exit notes at the time of this

writing, 19.74% of the total 4,737 relays on the Tor network. It is entirely possible than an adversay could control an exit node, giving them insight into the traffic leaving Tor. If the traffic is encrypted, the exit node could still see the DNS lookup and HTTP headers. If encryption is not implemented, all traffic would thus be available for inspection.

In September 2007, Swedish security consultant Dan Egerstad operated five exit nodes and monitored their traffic. Over the course of his experiment, he obtained the usernames and passwords for over 100 email accounts, read the correspondence from embassies in Australia, Japan, Iran, India, and Russia, along with communication between dignitaries including the Indian ambassador to China, various politicians in Hong Kong, workers in the Dalai Lama's liaison office, and several human-rights groups in Hong Kong. He also obtained sensitive spreadsheets and documents about military and political activities. Egerstad's analysis further revealed that approximately 95% of the traffic flowing over his nodes was not encrypted.[3]

An adversary could operate one or more exit nodes, as Egerstad did, or they could wiretap the exit's traffic and gain access to the traffic that way, perhaps without the knowledge of the exit operator. While the exit has no knowledge of the entire circuit and thus cannot deduce the origin of the traffic, the unencrypted traffic can be quite revealing and problematic, as Egerstad demonstrated. The primary way that Tor combats this through encryption. The Tor Browser Bundle comes with the HTTPS Everywhere, which prefers the use of HTTPS when connecting to web servers. Most email servers also support SSL/TLS, and users can also manually encrypt their documents or emails using PGP. Although Tor has no way of enforcing this final layer of encryption, it highly recommends it and places notices about this on many places on its website.

E. Web server vulnerabilities

The web server can also be a point of failure for the privacy and anonymity of the end user.

(going to expand this a bit more. Does the web server support SSL/TLS? The adversary could fake masquerade as the server)

F. Cryptographic Attacks

Although it is popular, a number of practical attacks and weaknesses have been demonstrated against its Key Scheduling Algorithm and PRNG that call into question RC4's security. In November 2013, Microsoft recommended disabling RC4 from TLS wherever possible.

G. Active Digital Watermarks

Another potential method to de-anonymize users is through active watermarking of packets. Digital watermarks require less time deploy, often have a higher success rate, and do not require observing traffic for long periods of time. If an attacker can modify the traffic flow such that it is noticable at the other end of Tor, then it becomes easy to identify end users and correlate them with their activities.

In 2011, Zhen Ling *et al* described an active watermark attack in which an attacker operates a reverse proxy and actively manipulates the size of the TCP packets transmitted to the exit node. Simultaneously, the attacker also runs a recovery method to detect the watermark on the user side. His watermark was a short sequence of 1s and 0s that could be encoded into the packet patterns. To transmit a 0, the attacker sends 498 bytes worth of web traffic to the Tor exit node, which is the exact size of the data portion of a Tor cell. Combined with the cell headers, the data packs into a single Tor cell of 512 bytes. To transmit a 1, the attacker sends 2444 bytes of data to the exit. The standard Maximum Transmission Unit (MTU) for Ethernet v2 lines is 1500 bytes and the TCP and IP headers are 20 and 32 bytes respectively, which means that a TCP packet can carry at most 1448 bytes of data. As the 2444 bytes from the attacker exceeds the MTU, two TCP packets are needed, one with a 1448-byte payload and the other with a 996-byte payload. The first TCP packet is received by the relay and is packed into three equally-sized cells, with padding added to the 452 bytes of the third cell. The second TCP packet becomes two cells of 498 bytes each, and thus the 2444 bytes becomes five Tor cells. Then these cells are processed, encoded as TCP packets, and retransmitted to the next relay. However, due to the design of cell queuing algorithms, TCP retransmission, and MTU restrictions, the two TCP packets may be split and combined into Tor cells in up to five different ways, each leaving a discernible packet signature. The attacker can then monitor the traffic near the client, examining the packet patterns generated by the attack. If the embedded signal is found, then it is quite likely that the traffic flows are correlated. In their tests, Zhen Ling *et al* achieved a detection rate of nearly 90%, with a false positive rate of less than 4%, concluding that such an attack could degrade anonymity.[11]

IV. Adversaries

Tor is designed to counter traffic analysis and circumvent censorship, so naturally most adversaries against Tor are those who wish to enforce Internet censorship or watch the electronic activities of certain groups and organizations. Such adversaries can range from a small number of individuals to whole branches within national governments.

A. China

One of the most famous examples of Internet censorship is the Great Firewall of China, a project launched in late 2003 which aims to prohibit individuals from using the Internet for activities which are not approved by the Chinese government. The Chinese Ministry of Public Security maintains the most sophisticated content-filtering regime in the world and actively engages in DNS cache poisoning, IP/URL blocking, and packet filtering.

Chinese citizens have increasingly grown increative in their attempts to *fanqiang*, or "climb the wall". The Chinese government responds by actively blocking VPNs, proxies, and Tor relays. Despite these efforts, users are able to bypass the firewall via *Tor bridges*, which act as the first hop into the

Tor network. The list of bridges is not publicly published, making them difficult to blacklist. The Great Firewall of China attempts to dynamically block these bridges by detecting the Tor TLS handshake, probing the target bridge, and then blocking it if it responds with the Tor protocol. However, users can still easily gain access to Tor (and thus the rest of the Internet) by the use of Obfsproxy. Bridges that are wrapped with the Obfsproxy tool disguise their TLS handshake so that it looks like normal HTTP traffic and otherwise carries no discernible byte characteristics to the outside world, making Obfsproxy bridges nearly impossible to detect.[14]

B. NSA

Earlier this year, Edward Snowden revealed that the United States National Security Agency (NSA) has been attempting to de-anonymize Tor users, but with little success. In a June 2012 PowerPoint presentation titled "Tor Stinks", the NSA stated that "with manual analysis we can de-anonymize a *very small fraction* of Tor users" but admitted that they have had no success in identifying a user on demand, and that they will never be able to de-anonymize all Tor users all the time. However, they had performed research on a variety of attacks against Tor. Their purpose was reveal "terrorists" and other individuals of interest to the United States.

Their documents revealed that as of 2012 they had access to very few nodes and that their goal is to control more, as they had only seen negligible success with circuit construction attacks. Further notes showed that they have had little success with timing attacks, and that they had put no effort towards revealing hidden services. While the NSA has Computer Network Exploitation (CNE) tools at their disposal for Windows, OS X, and Linux, as of 2012 CNE has been unsuccessful against the Tor Browser Bundle. If CNE exploits were possible, the NSA was considering the implications of attempting to direct the client to prefer NSA-controlled exit nodes, a private Tor network run by the NSA, or to reveal the client's IP by contacting an NSA web server directly. They were working on implementing this shaping attack. The NSA documents also showed that it was likely not legally or technically possible for them to exploit existing nodes, though they were considering exploiting web servers to make web surfing more difficult for Tor users. Considering psychological attacks, the authors concluded that Tor contained a critical mass of users, and that attempting to scare the public away from Tor could be counterproductive.[16]

The Tor developers responded to these and other related articles, saying "the good news is that they went for a browser exploit, meaning there's no indication they can break the Tor protocol or do traffic analysis on the Tor network. Infecting the laptop, phone, or desktop is still the easiest way to learn about the human behind the keyboard." and "if you attack too many users, somebody's going to notice. So even if the NSA aims to surveil everyone, everywhere, they have to be a lot more selective about which Tor users they spy on. Just using Tor isn't enough to keep you safe in all cases.". The developers also commented that the described attacks seemed "relatively basic" and "out-of-date", saying "we still have a lot of work to do to make Tor both safe and usable, but we don't have any new work based on these slides".[17]

C. FBI

One of the most famous attacks against Tor by the FBI was the successful finding and subsequent takedown of the Silk Road, a black market operating purely inside the Tor network as a hidden service. The Silk Road, sometimes called the "eBay of illegal drugs", first went online in February 2011, and remained covertly in use for two years later before Australian police discovered clues to its existence. The service could not be accessed from the traditional Internet, and, like other hidden services in the "darknet", was only available through the Tor network. On October 2, 2013, the FBI shut down the Silk Road and arrested the alleged owner, Ross Ulbricht in San Fransisco. Ulbricht mistakenly revealed his primary email address on a forum and later exposed his real IP by posing a question related to Silk Road on StackOverflow.com without using Tor. These clues then led to his arrest.

The Tor Project responded to the Silk Road takedown with an analysis of the Tor network, saying "So far, nothing about this case makes us think that there are new ways to compromise Tor (the software or the network). The FBI says that their suspect made mistakes in operational security, and was found through actual detective work." further saying that they found no evidence that the FBI broke into the webserver. They also reinforced their recommendations against providing personal information online.[15]

V. Analysis

(going to expand this a bit more and devote the couple pages to this section. I've got IEEE citations like [8], [11], and [12] that should be very helpful in demonstrating some weaknesses of Tor

Bauer and McCoy, *et al*, described a low-resource routing attack against the Tor network.[5]

VI. Conclusion

(obviously I need this. Summarize design, attacks, defenses, analysis, reviews in the literature, and then provide my opinion about the future)

References

[1] Matthew Wright, Micah Adler, Brian Neil Levine, and Clay Shields *The Predecessor Attack: An Analysis of a Threat to Anonymous Communications Systems*. ACM Transactions on Information and System Security (TISSEC) 4(7), November 2004, pages 489-522

[2] Abdelberi Chaabane, Pere Manils, Mohamed Ali Kaafar *Digging into Anonymous Trafc: a deep analysis of the Tor anonymizing network*. IEEE, 2010

[3] Kim Zetter *Rogue Nodes Turn Tor Anonymizer Into Eavesdropper's Paradise* Wired, 2007

[4] Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid *Recommendation for Key Management, Part 1: General (Revised)* National Institute of Standards and Technology, March 2007

[5] Kevin Bauer, Damon McCoy, Dirk Grunwald, Tadayoshi Kohno, Douglas Sicker *Low-resource routing attacks against tor*. ACM, 2007

[6] Damon McCoy, Kevin Bauer, Dirk Grunwald, Tadayoshi Kohno, Douglas Sicker, *Shining Light in Dark Places: Understanding the Tor Network*. Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2969, 2008.

[7] Nikoli, Ivica *Distinguisher and Related-Key Attack on the Full AES-256* Advances in Cryptology, 2009
[8] Liu Xin, Wang Neng *Design Improvement for Tor Against Low-Cost Traffic Attack and Low-Resource Routing Attack* 2009 International Conference on Communications and Mobile Computing
[9] National Security Agency *The Case for Elliptic Curve Cryptography*. NSA, 2009
[10] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger *Biclique Cryptanalysis of the Full AES* Microsoft Research Redmond, 2011
[11] Zhen Ling *Equal-Sized Cells Mean Equal-Sized Packets in Tor?* IEEE International Conference on Communications, 2011
[12] Zhen Ling, Junzhou Luo, Wei Yu, Xinwen Fuc, Weijia Jia, Wei Zhao *Protocol-level attacks against Tor* Computer Networks, 2012
[13] Robert Graham *Tor is still DHE 1024 (NSA crackable)*. Errata Security, 2013
[14] Daniel Anderson *Splinternet Behind the Great Firewall of China: The Fight Against GFW*. Association for Computing Machinery (ACM), Vol. 10, No. 11 (29 November 2012), doi:10.1145/2390756.2405036
[15] arma *Tor and the Silk Road takedown* The Tor Blog, 2 October 2013
[16] The Guardian *'Tor Stinks' presentation* The Guardian, 4 October 2013
[17] Roger Dingledine *Yes, we know about the Guardian article* Tor Project Blog, 4 October 2013
[18] Overview of Tor