A-Tor: Accountable Anonymity in Tor

Quanwei Cai[1,3], Jonathan Lutes[2], Jingqiang Lin[1,3,4], and Bo Luo[2(✉)]

[1] State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences,
Beijing 100093, China
[2] Department of Electrical Engineering and Computer Science,
The University of Kansas, Lawrence, KS 66045, USA
[3] Data Assurance and Communications Security Research Center,
Chinese Academy of Sciences, Beijing 100093, China
[4] School of Cyber Security, University of Chinese Academy of Sciences,
Beijing 100049, China

Abstract. Tor is the most popular anonymous communication system. In Tor, each user chooses onion routers (ORs) to construct a circuit to relay the traffic. The final OR of the circuit, called exit node, forwards regular traffic for the Tor user to the destination. As a result, the exit nodes are often accused of the anonymous users' illegal activities. In this paper, we propose an extension for Tor, called A-Tor, to provide accountable anonymity. A-Tor protects the exit nodes with verifiable evidences that the illegal or malicious packets are originated from the certain users but not the exit nodes. An A-Tor user firstly constructs a Tor circuit to apply for an anonymous certificate. Then, a second Tor circuit is constructed to access the destination server as in Tor, and the anonymous certificate is presented as a credential to the exit node; otherwise, the exit node refuses to forward his/her packets. A-Tor provides anonymity with the same level of assurance as Tor, and cooperative ORs are able to trace the anonymous A-Tor user (when illegal or malicious packets are detected in the future). Moreover, non-repudiation is achieved in the revocation of anonymity; that is, during the application of anonymous certificates and the subsequent anonymous communications through Tor circuits, a chain of evidences are generated by the A-Tor user and the ORs, and these evidences cannot be forged by collusive ORs. The performance overhead introduced by the A-Tor extension is also evaluated.

Keywords: Tor · Accountability · Revocable anonymity

1 Introduction

Tor is the most popular anonymous communication system in the Internet [11]. Anonymity is critical for personal privacy, but the Internet does not provide anonymity by default. So several anonymity networks such as Tor [10], Mixminion [8], Mix-master [19], and PipeNet [4,7], are designed and implemented to

X. Lin et al. (Eds.): SecureComm 2017, LNICST 238, pp. 838–851, 2018.
https://doi.org/10.1007/978-3-319-78813-5_46

unlink a communication party from his/her network activities. Tor balances anonymity, usability and efficiency well, and is deployed all over the world. Currently there are more than 2,000,000 Tor users, and the peak number is nearly 6,000,000 in 2013 [23].

A Tor user chooses a sequence of onion routers (ORs) to construct a circuit, and each OR in the circuit only knows its predecessor and successor. Encrypted packets sent by the user are wrapped by a symmetric key at each OR, and the final OR (or the *exit node*) forwards plaintexts IP packet to the destination server. Network packets from the destination server are iteratively encrypted by each OR and relayed to the next node, and the Tor user finally decrypts the packets with all symmetric keys.

Due to its excellent anonymity and general availability, Tor is misused to launch network attacks. As a result, the exit nodes are often accused of the anonymous Tor users' illegal network activities by law enforcement agencies. For example, Rapid7 revealed a botnet called SkyNet, which adopts Tor for the command-and-control communications [2]. Austrian seized computers from the owner running a Tor exit node because cyber crimes were committed through this exit node, and announced that it is illegal to run Tor exit nodes [3]. Moreover, the insufficient protection and possible liability burden of Tor exit nodes discourage volunteer ORs to be exit nodes; then, if there are only a very limited number of exit nodes, it becomes easier to associate an anonymous Tor user with his/her network packets by traffic analysis [10,21].

With the original anonymity functionality of Tor, it is extremely difficult for the exit nodes to prove that the IP traffic is originated from other nodes; otherwise, the anonymity will be degraded somehow. Some extensions for Tor are proposed to protect the exit nodes, by enforcing exit policies [10,22] or appending specific packet headers [10]. These extensions offer options for exit nodes and ORs, but such packet headers are not verifiable and cannot disclose the Tor user's identity such as its IP address. Trusted third parties are introduced [6,15] to escrow the Tor user's identity before the anonymous communications and revoke the anonymity when necessary; however, the extra trust on the third party degrades the anonymity of Tor, because a single compromised party is able to reveal the user's identity. A reputation system is designed for exit nodes to rank the anonymous Tor user's activities [12], and the users with low reputation will be marked. This scheme depends on the intrusion-detection capability of exit nodes, and brings a significant overhead to the exit node.

In this paper, we propose *A-Tor*, *accountable anonymity* in Tor, which protects exit nodes with *verifiable evidences* to revoke the anonymity of Tor users. A-Tor designs a two-phase protocol. In the first phase (called the *anon-cert* phase), an A-Tor user firstly constructs a Tor circuit to apply for an anonymous certificate from the last OR (called the *certification node* in this paper). A chain of evidences is generated by the user and the ORs during the application, and these evidences will be used to trace the A-Tor user based on the anonymous certificate. Then, in the second phase (called the *anon-comm* phase), a second Tor circuit is constructed to access the destination server, and this certificate is

presented as a credential to the exit node; otherwise, the exit node refuses to forward his/her IP packets to the destination. The forwarded packets are signed by the anonymous user, and verified by the exit node using the anonymous certificate before sent to the destination server.

In summary, A-Tor achieves accountable (or revocable) anonymity with the following properties:

- It is built on top of Tor, and the anonymity of Tor is not degraded. In the anon-cert phase, the anonymous certificate is generated through a Tor circuit, with the same level of assurance as Tor. The anonymous certificate is visible only to the A-Tor user, the certification node and the exit node. In the anon-comm phase, the certificate is presented as an anonymous credential to the exit node, and no any other identity information is transmitted on the second Tor circuit. The anonymity would be broken, only if (a) the ORs of the anon-comm Tor circuit, or (b) the ORs of the anon-cert Tor circuit and the exit node of the anon-comm Tor circuit, collude to link the A-Tor user to his/her network activities.
- Non-repudiation is achieved in the revocation of anonymity. In the application of anonymous certificates, a chain of evidences are generated by the A-Tor user and the ORs, and each evidence is signed by the generator and sent to in the next node of the anon-cert Tor circuit. During the anonymous communication, the network packets to the destination server are signed and verified by the exit node using the anonymous certificate. Therefore, these evidences are also verifiable to law enforcement agencies, and could not be forged by malicious ORs cooperatively against an innocent user.
- A-Tor is an extension of Tor, and interoperable with the existing Tor ORs. No additional component is needed in A-Tor, compared with Tor. A-Tor extension functions are implemented by Tor ORs, and transparent to the destination servers. An anonymous user may enable the A-Tor extension or use the original version of Tor, to construct the anonymous communication circuits. Then, the exit node chooses to forward or reject the packets, according to its own policy.

2 Background and Related Work

Various schemes are proposed to protect the exit nodes in Tor. [10,22] provide mechanisms for exit nodes to limit the relayed traffic. That is, each node may specify its exit policy to describe the addresses and ports that it will connect to; the exit node uses port restrictions for certain services (e.g., HTTP, SSH and FTP). However, it does not provide a complete protection for exit nodes, as most of the abuse cases are based on the protocols widely supported. [10] allows an OR to add specific information in the header of the forwarded messages, to indicate that the traffic is originated from some users of the anonymity service. However, the auditor cannot distinguish whether the traffic was truly originated from an anonymized user, or from a malicious exit node which added fake header

attributes to its own messages. A reputation system is built based on the activities of anonymous users, and the exit node may reject an anonymous user based on its history [12].

Different accountable anonymity schemes in Tor have been proposed by introducing trusted third parties. [6] requires a trusted party to generate a blind signature as the ticket for the anonymous user to access the anonymity services, and the user's anonymity is revoked with the ticket and the trusted party. The ticket plays the similar role as the anonymous certificate in A-Tor, but the trusted party is able to reveal the user's identity [6] while the certification node in our scheme cannot without the cooperation of ORs in the anon-cert circuit. The directory server is utilized as the verifier of message transmitted from anonymous Tor users [15]. A Tor user divides its IP address into multiple shares, and the directory server signs a ticket for the shares and the hash value of messages to be transmitted. Then, the IP address shares are distributed to Tor ORs, and the signed ticket is presented as a credential to relay the messages. The IP address shares are collected to revoke the user's anonymity when necessary. These two schemes degrade the anonymity of Tor, as the trusted party or the directory server may collude with the exit node to break the users' anonymity.

Different mechanisms are also proposed to incentivize ORs to relay Tor traffic. When a well-behaving OR, acts as a Tor user to construct a circuit, the traffic of this circuit will be relayed with higher priority [20]. BRAIDS [14] motivates anonymous users to relay Tor traffic by introducing generic tickets for service accounting. The ticket is generated using blind signatures, which ensures the ticket signers do not know the ORs chosen by the user. These schemes work compatibly with A-Tor.

Anonymous blacklisting schemes [17,18,24,25] are proposed to prevent future abusive anonymous access. These schemes are classified into two classes: one [17, 25] depends on trusted third parties to provide tokens for users to access the service providers (i.e., destination servers), while in the other schemes [18,24], each user presents the proof that it is not blacklisted. However, the identities of abusive users are not revealed in these schemes.

There are also revocable anonymity schemes, not designed for Tor. In [9], each user registers with the trusted authority and a chosen registration node, to link its unique identifier to an identification pseudonym, and the identification pseudonym to another pseudonym for anonymous services. The two pseudonyms are verifiably encrypted using the public key of Judge, who identifies the initiator of the malicious traffic, based on the information from the exit node of the anonymity network, the register node and the trusted authority. THEMIS [26] relies a trusted key generator to achieve accountable anonymity and non-frameability based on proxy re-encryption. The trusted key generator who does not know the user's identifier, distributes an anonymous certificate and the corresponding index to the user and the identity database, respectively. The cooptation between the trusted key generator and the identity database will combine the anonymous certificate with the user's identity.

In [16], each user firstly connects to a group of servers (called anonymisers) to obtain an encrypted identity, and applies for an anonymous certificate signed by blind signature algorithms to bind the encrypted identity to a key pair that is used in anonymous communications. When necessary, a threshold atomic proxy re-encryption is triggered at the chosen anonymisers to transfer the user's identity encrypted using the auditor's public key. Compared with [16], A-Tor also utilizes anonymous certificates to support accountable anonymity; but A-Tor seamlessly integrates the application of anonymous certificates into Tor so that the anonymity is evaluated explicitly.

To protect Tor gainst abuse by botnets, one possible medium-term response is to deanonymize the command and control (C&C) server [13]. The Tor Project attempts to discover the entry nodes of the C&C server, by repeatedly changing their availability (e.g. by rotating identity keys), and eventually learn the IP address of the C&C server. However, it will pose significant organizational and engineering challenges [13], while A-Tor finds the malicious Tor user with fewer overhead as described in Sect. 4.3.

3 Overview of A-Tor

3.1 Threat Model and Design Goal

A-Tor follows the same assumptions as Tor [10]. A great number of ORs run over the Internet, each of which maintains a key pair by itself. The public key of an OR is published in directory servers, and then known by all users and other ORs. A correct OR follows the protocol strictly and compromised ORs behave arbitrarily. We assume that the ORs of a Tor circuit are not compromised simultaneously; so a user will increase the number of ORs in a circuit, to enhance the assurance level of anonymity.

A-Tor attempts to prevent attackers from linking a pair of communication parties (i.e., an A-Tor user and the destination server) or from linking multiple communications to or from a single user as Tor does, while provides verifiable evidences to link (the packets of) a specified communication to an anonymous user when enough ORs cooperate. The anonymity of an A-Tor user is compromised only if a certain number of ORs are compromised to link his/her activities, and this number is specified by each user according to his/her own security concern. These evidences are stored on multiple ORs for the period of time specified in data retention laws, and presented together to reveal the user's identity of a specified communication when a law enforcement agency requires the ORs to do. Moreover, malicious ORs could not collude to forge a complete chain of evidences against an innocent user.

Finally, because A-Tor attempts to provide verifiable and unforgeable evidences to reveal the user identity, we assume that each A-Tor user has an identity credential (e.g., a non-anonymous X.509 certificate to certify his/her IP address or other alternative identity), which is verifiable to the ORs. More discussions about this credential are included in Sect. 6.

3.2 Basic Idea

The basic idea of A-Tor is, (*a*) in the anon-cert phase, an A-Tor user constructs a Tor circuit to apply for an anonymous certificate from the last OR (or the certification node), and (*b*) in the anon-comm phase, the anonymous certificate is presented as a credential to the exit node of the second Tor circuit.

In the anon-cert phase, the A-Tor user constructs the anon-cert Tor circuit. Then, the user sends the credential of his/her identity to the first OR of the anon-cert Tor circuit (called the *registration node*). After verifying the credential, the OR signs an anonymous-certificate request, encrypts the message by the public key of the next OR of the circuit, and sends it. Then, after decrypting the message and verifying the signature, the receiver OR signs, encrypts and sends it to the next OR, until the anonymous-certificate request is transmitted to the certification node. Finally, the certification node signs the anonymous certificate, and it is relayed to the A-Tor user. The certificate is encrypted iteratively by each OR as regular Tor packets. Note that the A-Tor's identity is not included in either the anonymous-certificate request or the certificate. The signed request messages are stored on the receiver ORs as verifiable evidences to reveal the A-Tor user's identity in the future.

Next, after constructing the anon-comm circuit as Tor, the A-Tor user sends the anonymous certificate to the exit node, and the exit node verifies that the certificate is signed by another OR. Then, each relayed packet is signed by the A-Tor user, and the exit node verifies the signature using the anonymous certificate before forwarding it to the destination server. These signatures are stored on the exit node as verifiable evidences. The anonymous certificate and the signatures are invisible to other ORs of the anon-comm Tor circuit. Signing every packet one by one is expensive, and optimizations are discussed in Sect. 6.

No additional component is needed in A-Tor, compared with Tor. Each OR of A-Tor is first an OR of Tor, and the A-Tor functions are extended on the ORs of the anon-cert Tor circuit and the exit node. Anonymous-certificate request messages and signed network packets are transmitted by extended commands in the Tor circuit [10].

4 The A-Tor Protocol

This section describes the A-Tor protocol in details, including the steps to apply for anonymous certificates, to perform anonymous communications, and to link the anonymous communication to the A-Tor user.

These notations are used in this paper:

- ID_i^{OR}, PK_i, SK_i: the identity, the public key and the private key of OR_i.
- PK_u, SK_u: an ephemeral key pair generated by the user.
- ID_i^c: the connection identity between the user and OR_i in the Tor circuit.
- $Enc_K[e]$, $Dec_K[e]$: encrypt and decrypt message e by key K.
- $Sign_K[e]$: sign message e by key K.

4.1 Anonymous Certificate

As shown in Fig. 1, the A-Tor user constructs the anon-cert Tor circuit consisting of m ORs, denoted as OR_i and $1 \leq i \leq m$. OR_1 is the registration node, and OR_m is the certification node. After the circuit is constructed, the user shares a secure connection with OR_i, which is identified as ID_i^c. The detailed steps to construct a Tor circuit is described in [10].

Then, the user generates an ephemeral key pair (PK_u, SK_u), constructs an anonymous certificate request $ACertReq_{m+1} = Sign_{SK_u}[PK_u, T_s, T_e]$, and computes $ACertReq_i = ID_i^{OR}||Enc_{PK_i}[ID_i^c, ACertReq_{i+1}]$ iteratively, where (T_s, T_e) is the period of validity and $1 \leq i \leq m$. The certificate requests are also encrypted like the layers of an onion. Next, the user sends $ACertReq_1||cred$ to OR_1, where $cred$ is a credential of his/her identity.

Upon receiving $ACertReq_1||cred$, OR_1 verifies $cred$ and decrypts it to obtain $ACertReq_2$. Then, it sends $Sign_{SK_1}[ACertReq_2]$ to OR_2. Upon receiving $Sign_{SK_{i-1}}[ACertReq_i]$, OR_i verifies the signature and decrypts it to obtain $ACertReq_{i+1}$, until OR_m obtains $ACertReq_{m+1}$. At the same time, OR_i stores $Sign_{SK_{i-1}}[ACertReq_i]$ as a verifiable evidence.

OR_m signs the certificate $ACert_{u,m} = Sign_{SK_m}(PK_u, ID_m^{OR}, T_s, T_e)$. This anonymous certificate is relayed back to the user through the Tor circuit, encrypted iteratively by ORs.

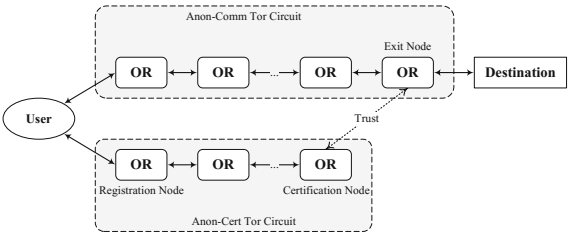

Fig. 1. The A-Tor protocol.

4.2 Anonymous Communication

The steps to perform anonymous communication in A-Tor is almost the same as those in Tor, except that the relayed network packets are signed by the A-Tor user and verified by the exit node.

As shown in Fig. 1, an A-Tor user constructs a Tor circuit consisting of n ORs, denoted as OR_j and $1 \leq j \leq n$. OR_n is the exit node, and no OR is in the anon-cert Tor circuit and the anon-comm Tor circuit at the same time. After the Tor circuit is constructed and before any packet to the destination is sent, the A-Tor user sends $ACert_{u,m}$ to the exit node through the circuit. The exit node verifies that the certificate is signed by another OR, and in its period of validity, and replies with an acknowledgement. Otherwise, it rejects to forward any packet.

Then, the user begins to send packets through the Tor circuit. It signs every packet using SK_u and transmits the signed packet to the exit node. The exit node forwards a packet (without the signature) to the destination server, only if it is sent along with a valid signature. The Tor circuit shall be closed before the anonymous certificate expires. The certificate and signatures are stored by the exit node as verifiable evidences in the future.

4.3 Accountable Anonymity

If the traffic forwarded by the exit node is detected to be illegal or malicious, an auditor who is authorized by law enforcement agencies, performs the following steps to reveal the user's identity.

The auditor brings the illegal or malicious packets to the exit node, and the exit node will present the corresponding certificate $ACert_{u,m}$ and signatures. After verifying the certificate and signatures, the auditor requires OR_m to present $Sign_{SK_{m-1}}[ACertReq_m]$. Otherwise, if the certificate or any signature is invalid, the exit node is liable for these illegal or malicious packets.

OR_m decrypts $ACertReq_{m+1}$ from $ACertReq_m$, and the auditor checks whether the anonymous-certificate request message $ACertReq_{m+1}$ matches the certificate $ACert_{u,m}$ or not; if they match, the auditor verifies the signature by OR_{m-1} and then requires OR_{m-1} to present $Sign_{SK_{m-2}}[ACertReq_{m-1}]$.

The auditor finds out the ORs one by one in the anon-cert Tor circuit and finally the registration node presents $ACertReq_1||cred$. The user's identity is revealed in $cred$ verifiably. In the above steps, if any OR cannot present a valid certificate request message, the OR is liable for these illegal or malicious packets.

5 Security Analysis and Performance Evaluation

This section analyzes the accountable anonymity of A-Tor. We first evaluate the assurance level of anonymity, the verifiable evidences to reveal the A-Tor user's identity, and the performance overhead of A-Tor. Finally, some optimizations and extended discussions are presented.

5.1 Anonymity

A-Tor provides anonymity with the same level of assurance as Tor. Firstly, in the anon-comm phase, all steps of A-Tor are the same as those of Tor, except that an anonymous certificate is transmitted to the exit node. Because there is no identity in the anonymous certificate and no OR of the anon-comm Tor circuit is involved in the steps to apply for anonymous certificates, attackers cannot obtain more information than a Tor circuit to break the anonymity. Secondly, no (anonymous) communication with destination servers is performed in the anon-cert phase, so attackers cannot obtain any information by only compromising the ORs of the anon-cert Tor circuit.

Next, let's consider the scenario that some ORs of two Tor circuits were compromised, and assume that the certification node runs independently of the exit node. Because the certificate requests are also encrypted like the layers of an onion, only when all ORs of the anon-cert Tor circuit collude, they reveal the user's identity and link it to the anonymous certificate; and only the exit node is able to link the anonymous certificate to the network activities. So, only if all ORs of the anon-cert Tor circuit and the exit node of the anon-comm circuit are compromised, they are able to collude to link the A-Tor user to his/her activities.

Compared with Tor, there are two sequences of ORs that are able to link the communication to an A-Tor user: one is composed of the ORs in the anon-comm Tor circuit, and the other is the ORs of the anon-cert Tor circuit and the exit node. Therefore, $m = n - 1$ is reasonable (provided that the certification node runs independently of the exit node), and two sequences establish equal difficulties for attackers to break the anonymity. Moreover, in the second sequence, each OR only knows its predecessor and successor as that in the first sequence, except that the certification node does not know its successor (i.e., the exit node). Note that the anonymous certificate is transmitted as ciphertext always to the exit node through two Tor circuits.

As for other passive attacks, active attacks and directory attacks, A-Tor provides the same protections as Tor [10]. A-Tor constructs two Tor circuits to generate verifiable evidences for accountable anonymity, and the accountable anonymous communications are wrapped as regular Tor packets.

A-Tor does not introduce additional traffic patterns, compared other application protocols on top of Tor. Passive attackers cannot distinguish an A-Tor user from Tor users, because the additional anon-cert phase works over regular Tor circuits. Active attacks do not have more attack opportunities, because the ORs in A-Tor do not have more security assumptions than Tor. Each OR holds its key pair, and only know its predecessor and successor in the two Tor circuits. Finally, directory servers maintain the same information as those in Tor.

5.2 Verifiable Evidences

The verifiable evidences are composed of: (a) the certificate and signatures stored on the exit node, and (b) the certificate requests stored on the ORs of the anon-cert Tor circuit. Section 4.3 shows that, the A-Tor user's identity will be revealed, if the auditor follows the evidences to find out the ORs one by one.

Next, we will show that, (a) nobody can forge such a chain of evidences, unless the A-Tor user and ORs involved in the accountable-anonymous communications, and (b) the ORs cannot misguide the auditor to innocent ORs or users, either intentionally or unintentionally.

As all evidences are signed messages, nobody can forge these evidences unless a private key was compromised. In particular, in the trace path to reveal the A-Tor user's identity, the exit node is located by the IP address of packets. Then, the signatures of forwarded packets and the anonymous certificate are signed by the A-Tor user and the certification node, respectively. The anonymous-

certificate requests are signed by the ORs one by one in the anon-cert Tor circuit, and $ACertReq_1$ is sent along with *cred*, which is also verifiable and unforgeable.

When OR_i decrypts $ACertReq_{i+1}$ from $ACertReq_i$, a malicious OR might present an unrelated anonymous-certificate request signed by OR_k, but intentionally output $ACertReq_{i+1}$. Note that, the private key of OR_i shall not be disclosed to the auditor, so the decryption is performed by OR_i itself. Then, the auditor will mistakenly require OR_k instead to present valid certificate request messages, and OR_k will be liable for the illegal or malicious packets for it is unable to do so. Therefore, the public-key encryption algorithm shall be deterministic but not probabilistic, and the auditor needs to check whether the plaintext (i.e., $ACertReq_{i+1}$) and the ciphertext (i.e., $ACertReq_i$) match or not. If a probabilistic public-key encryption algorithm such as RSA or ECIES, is adopted, $ACertReq_i$ is revised to $ID_i^{OR}||Enc_{K_i}[ID_i^c, ACertReq_{i+1}]||H(K_i)||Enc_{PK_i}[K_i]$ and signed by OR_{i-1}, where $H()$ is a one-way hash function and K_i is a one-time session key of symmetric encryption algorithms. Therefore, when O_i outputs K_i, the auditor firstly checks whether K_i and $H(K_i)$ match or not and then decrypts $ACertReq_{i+1}$ by itself.

5.3 Performance Evaluation

We evaluated the performance overhead introduced by the A-Tor extension, by measuring the average processing time for the circuit establishment and the network packet relay. The same as the lastest version of Tor [1], we adopt ECC-Curve25519 [5] and SHA-256 for key negotiation. In details, we use the implementation of Curve25519 and Ed25519 in Tor for key negotiation and signature generation. For symmetric encryption and hash function, we adopt AES-128 and SHA-256 in OpenSSL v1.01f. The process in each node is implemented using C++. The numbers of ORs in the anon-cert Tor circuit and the anon-comm Tor circuit satisfy the equation $m = n - 1$. All experiments ran with one user. These nodes were deployed on the identical workstations with an Intel i7-3770 (3.4 GHz) CPU and 12 GB of memory. The operating systems of all the nodes are CentOS v6.6. We measured the average processing time by constructing a Tor circuit and sending a cell 100 times.

To construct a circuit, a Tor user negotiates a symmetric key with each OR in the circuit. The A-Tor user needs to construct two Tor circuits. The anon-comm Tor circuit is constructed as the original Tor circuit. The anon-cert Tor circuit construction includes the following processes: the A-Tor user negotiates the symmetric key with each OR, constructs the anonymous-certificate request and binds its credential (in our experiments, a signature of the transmitted message using its long-term private key); the registration node checks the user's credential and generates the signature of the transmitted messages; the other OR except the certification node verifies the received signature and generates a new one for its transmitted message; and the certification node constructs the anonymous certificate after verifying the received signatures. In our implementation, as described in Sect. 5.2, ID_i^c and $ACertReq_i$ are encrypted using the symmetric key shared with each OR, and the digest of the symmetric key is

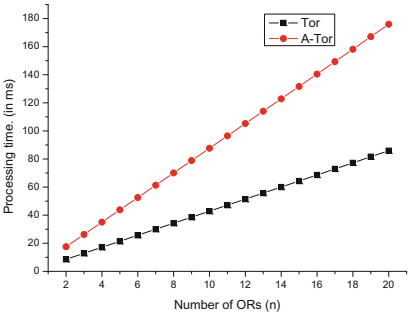

Fig. 2. Circuit establishment.

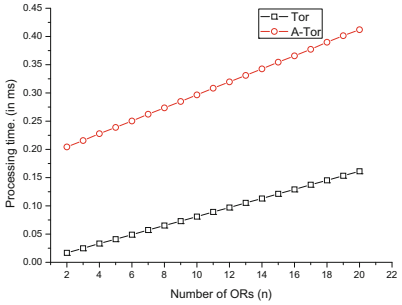

Fig. 3. Anonymous cell processing.

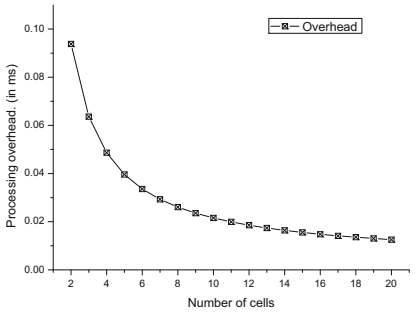

Fig. 4. Anonymous cell processing (accumulatively signed).

included in *ACertReq*. From Fig. 2, we find that when $n \leq 20$, the time to construct the two Tor circuits (the anon-cert circuit and the anon-comm circuit) is about 2.049 times of the one to construct the original Tor circuit.

After establishing the circuits, the user begins to send the network packets to destination servers. In Tor, the traffic is split into cells of 512 bytes, while the size of A-Tor cell is at most 498 bytes. In addition to AES encryption/decryption, each A-Tor cell involves two extra processes: the user generates the signature (64 bytes) of the cell, and the exit node verifies the signature. From Fig. 3, we find that the ratio of anonymous cell processing in A-Tor to that in Tor decreases with the number of ORs, from 12.207 ($n = 2$) to 2.555 ($n = 20$). The primary overhead is caused by the signature generation (0.05521 ms) and verification (0.13239 ms).

We adopt the optimization described in Sect. 6.1 to reduce the overhead of anonymous cell processing in A-Tor. That is, instead of generating and verifying the signature for each cell, we accumulatively compute the digest of these cells, generate and verify the signature of these digests. From Fig. 4, it is found that, the average overhead of A-Tor reduces with the number of cells for accumulative digests, and is reduced to 0.01254 ms when the number of cells is 20, which is modest compared to the processing time (0.01674 ms) for one cell in original Tor when the number of ORs is 2.

6 Extended Discussion

6.1 Signing and Verification in the Anonymous Communication

It is very expensive to sign and verify each packet one by one in the anonymous communication. The following optimizations of coarse-grained evidences are designed to reduce the overheads. Firstly, the A-Tor user may only sign a description file on his/her visit to the destination server. The description includes, for example, the destination server, the port, the duration, and the accessed web pages, but not any specific packets.

Or, after verifying the certificate, the exit node randomly sends some packet-signing commands through the Tor circuit to the A-Tor user during the anonymous communication. The A-Tor user signs the next packet, once it receives a signing request from the exit node. A portion of signed packets in the attack traffic shall be enough to play as verifiable evidences, and the frequency of signing requests is determined by the exit node.

Another optimized mode is as follows. The A-Tor keeps sending packets and the exit node forwards these packets while accumulatively computing the digest of these packets, until a threshold of sent-but-unsigned packets is triggered and the exit node sends a sign-all-packet command. Then, the A-Tor user signs all sent-but-unsigned packet as a whole, and sends the signature to the exit node. Next, unsign packets are forwarded again. The maximum count (or length) of sent-but-unsigned packets also depends on the policy of exit nodes.

6.2 Credential of the User's Identity

In Sect. 4.1, we assume that the A-Tor user has an identity credential verifiable to the registration node. A typical example is an non-anonymous X.509 certificate, and the A-Tor user signs the anonymous-certificate request as the verifiable credential. Or, the registration node cooperate with ISPs to verify the A-Tor user's identity.

Because A-Tor attempts to provide unforgeable evidences to reveal the user identity but the default identity in the Internet (i.e., IP address) can be forged, an extra trusted identity shall be presented to the registration node.

6.3 Key Revocation of ORs

The key pair (PK_i, SK_i) of OR_i might be revoked due to security incidents. The revoked key pair may be needed for the auditor to reveal the malicious user's identity. Therefore, the directory server should record the revoked public key and the corresponding period of validity correctly, while each OR maintains the corresponding private key. The storage period should be no less than the one specified in data retention laws.

An OR in the anon-certificate Tor circuit or the exit node, should check the validity of public keys when receiving certificate requests and anonymous certificates, and reject any message signed using a revoked key pair; otherwise,

it will be accused instead of the OR whose key pair has been revoked at the directory server.

7 Conclusion

In this paper, we propose A-Tor, an extension of Tor, protecting the exit nodes in Tor by verifiable evidences. An A-Tor user firstly applies for an anonymous certificate through a Tor circuit, and the anonymous certificate is used as credentials in another Tor circuit for the next anonymous communications to destination servers. A-Tor provides anonymity with the same level of assurance as Tor, and cooperative ORs are able to trace the anonymous user (when illegal or malicious packets are detected in the future). The Tor circuit of anonymous certificates does not cut down the attack difficulties to break the anonymity, and the same number of ORs shall be compromised as in Tor before the attacker links an A-Tor user to his/her network activities. A chain of verifiable evidences are generated during the application of anonymous certificates and the anonymous communications, and non-repudiation is achieved.

Acknowledgments. Q. Cai, and J. Lin were partially supported by National 973 Program of China under Award No. 2014CB 340603. B. Luo was partially supported in part by US National Science Foundation under NSF CNS-1422206, NSF DGE-1565570l.

References

1. Tor project. https://github.com/torproject/tor
2. Skynet, a Tor-powered botnet straight from Reddit (2012). https://community.rapid7.com/community/infosec/blog/2012/12/06/skynet-a-tor-powered-botnet-straight-from-reddit
3. Austrian Tor exit node operator found guilty as an accomplice because someone used his node to commit a crime (2014). https://www.techdirt.com/articles/20140701/18013327753/tor-nodes-declared-illegal-austria.shtml
4. Back, A., Möller, U., Stiglic, A.: Traffic analysis attacks and trade-offs in anonymity providing systems. In: 4th International Workshop on Information Hiding (IH), pp. 245–257 (2001)
5. Bernstein, D.J.: Curve25519: new Diffie-Hellman speed records. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 207–228. Springer, Heidelberg (2006). https://doi.org/10.1007/11745853_14
6. Claessens, J., Diaz, C., Goemans, C., Dumortier, J., Preneel, B., Vandewalle, J.: Revocable anonymous access to the Internet? Internet Res. **13**(4), 242–258 (2003)
7. Dai, W.: PipeNet 1.1. Technical report, Usenet Post (1996)
8. Danezis, G., Dingledine, R., Mathewson, N.: Mixminion: design of a type III anonymous remailer protocol. In: 24th IEEE Symposium on Security and Privacy (S&P), pp. 2–15 (2003)
9. Díaz, C., Preneel, B.: Accountable anonymous communication. In: Petković, M., Jonker, W. (eds.) Security, Privacy, and Trust in Modern Data Management, pp. 239–253. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-69861-6_16

10. Dingledine, R., Mathewson, N., Syverson, P.: Tor: the second-generation onion router. In: 13th Usenix Security Symposium, pp. 303–320 (2004)
11. Elahi, T., Danezis, G., Goldberg, I.: PrivEx: private collection of traffic statistics for anonymous communication networks. In: 21st ACM Conference on Computer and Communications Security (CCS), pp. 1068–1079 (2014)
12. Groš, S., Salkić, M., Šipka, I.: Protecting Tor exit nodes from abuse. In: 33rd International Convention MIPRO, pp. 1246–1249 (2010)
13. Hopper, N.: Protecting Tor from botnet abuse in the long term (2013). https://research.torproject.org/techreports/botnet-tr-2013-11-20.pdf
14. Jansen, R., Hopper, N., Kim, Y.: Recruiting new Tor relays with BRAIDS. In: Proceedings of the 17th ACM Conference on Computer and Communications Security (CCS), pp. 319–328 (2010)
15. Kane, A.M.: A revocable anonymity in Tor. Technical report, IACR Cryptology ePrint Archive (2015)
16. Köpsell, S., Wendolsky, R., Federrath, H.: Revocable anonymity. In: Müller, G. (ed.) ETRICS 2006. LNCS, vol. 3995, pp. 206–220. Springer, Heidelberg (2006). https://doi.org/10.1007/11766155_15
17. Lofgren, P., Hopper, N.: BNymble: more anonymous blacklisting at almost no cost (a short paper). In: Danezis, G. (ed.) FC 2011. LNCS, vol. 7035, pp. 268–275. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27576-0_22
18. Lofgren, P., Hopper, N.: FAUST: efficient, TTP-free abuse prevention by anonymous whitelisting. In: ACM Workshop on Privacy in the Electronic Society, pp. 125–130 (2011)
19. Möller, U., Cottrell, L., Palfrader, P., Sassaman, L.: Mixmaster Protocol - Version 2. IETF Internet-Draft (2004)
20. "Johnny" Ngan, T.-W., Dingledine, R., Wallach, D.S.: Building incentives into Tor. In: Sion, R. (ed.) FC 2010. LNCS, vol. 6052, pp. 238–256. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14577-3_19
21. Serjantov, A., Sewell, P.: Passive attack analysis for connection-based anonymity systems. In: Snekkenes, E., Gollmann, D. (eds.) ESORICS 2003. LNCS, vol. 2808, pp. 116–131. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39650-5_7
22. Syverson, P., Reed, M., Goldschlag, D.: Onion routing access configurations. In: DARPA Information Survivability Conference and Exposition (DISCEX), vol. 1, pp. 34–40 (2000)
23. The Tor Project: Tor metrics (2017). https://metrics.torproject.org/
24. Tsang, P.P., Au, M.H., Kapadia, A., Smith, S.W.: PEREA: towards practical TTP-free revocation in anonymous authentication. In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 333–344 (2008)
25. Tsang, P.P., Kapadia, A., Cornelius, C., Smith, S.W.: Nymble: blocking misbehaving users in anonymizing networks. IEEE Trans. Dependable Secure Comput. **8**(2), 256–269 (2011)
26. Xu, G., Aguilera, L., Guan, Y.: Accountable anonymity: a proxy re-encryption based anonymous communication system. In: 18th IEEE International Conference on Parallel and Distributed Systems (ICPADS), pp. 109–116 (2012)