Another Tor is possible

Amadou Moctar Kane

KSecurity, BP 47136, Dakar, Senegal. amadou1@gmail.com

October 2, 2014

Abstract

The aim of this paper is to introduce some modifications in Tor, in order to improve user's anonymity and relay's security. Thus, we introduced a system that will ensure anonymity for all users, while maintaining the ability to break the anonymity of a sender in case of misconduct. The revocation of the anonymity will require the use of secret sharing schemes, since we assume that, the lifting of the anonymity of the dishonest user should not depend on a single entity, but on a consensus within the network. In addition to the revocation of the anonymity, we propose in this paper further improvements such as mixing Tor traffic with those of the major internet groups, using the camouflage, or introducing a honeypot in the network.

Keywords : Tor, cryptography, privacy, anonymity, secret sharing.

1 Introduction

Recent revelations about the mass surveillance of populations will cause increased and disordered use of tools designed to protect the privacy of users. The most important tools which can provide privacy depend on networks anonymity; unfortunately these tools have become the haunt of drug dealers, pedophiles, and spies. It is important to note that those who agree to make their resources available to the Tor network are exposed to serious consequences (especially exit nodes), from the best case where it is an IP blocked and the worst case where they face legal or extrajudicial prosecution. Hence, this may be the reason why we have a very little number of onion router (OR) in the Tor network (about 6000 OR for around 300,000 users in the recent past).

In addition, Tor faces several censures because of its non-revocable anonymity [2], so it is urgent to clean this network to allow peaceful users, journalists and cyber dissidents to be able to do quietly their business on the Internet.

If a revocable anonymity is introduced, then it is certain that pedophiles and other criminals would go elsewhere, and governments that are blocking Tor, would no longer have a pretext to do so.

In addition, if Internet users know that, in case of misconduct of Tor users, they would not have to pay the piper, then it is likely that the number of exit node would increase significantly.

The anonymity can be defined as the quality or state of being unknown or unacknowledged.

Even if most comments on the Internet seems to be anonymous, the anonymity on the Internet implies that a user should not be identifiable through its network traffic any more than any other user of the internet,

that is, the network traffic should have no identifying characteristics [1]. In view of this definition, we can affirm that the anonymity is not present on the internet that we currently use, since the routing requires the specification of an IP address and a port destination. To solve this problem, significant research has been conducted in anonymous networks and most ideas rely on Mix Network.
The main idea of the Mix network consists on taking messages from multiple senders, shuffling them, and sending them back out in random order to the next destination. Hence, this process can break the link between the source of the request and the destination.
Later, the Onion Routing inspired by the Mix network emerge. It was introduced and patented by the U.S. Navy, as described in [3]: This protocol gets its name from its method of encrypting the initial packet and the address of the proxies at each hop on the path with the public key of the previous step. This scheme results in layers of encryption that are peeled off at each step in order to determine the next address to send to on the path. This requires the initiator to predetermine the entire path.
More recently, Tor was created from the Onion Routing principles. It was originally Funded by the United States Naval Research Centre, the United States Defense Advanced Research Projects Agency (DARPA) and supported by the Free Haven Project, Tor (The Onion Routing) is currently the largest anonymous network in existence [5].

1. Anonymity:

   In order to introduce a revocable anonymity in Tor, in this paper, we consider the revocation of the anonymity as the fact of showing the IP address of the offender. The revocation of an anonymity in Tor must obey some rules, because if it depends on only one entity, this one could be blackmailed or victim of "legal", or illegal pressure.

2. Secret sharing schemes:

   One of the two first secret sharing schemes was introduced by Adi Shamir [4]. In a threshold secret sharing scheme, the dealer is the ones who divides the secret, into multiple parts called shares, which it distributes among a set of participants. There is a quorum of shares required to reconstruct the original secret back, but fewer than the threshold number of shares can't recover any information about the secret. The threshold secret sharing scheme is secure if the dealer and the participant are honest, that is why a verifiable secret sharing scheme was introduced.

   In this paper, we take the novel approach (definition) for verifiable secret sharing presented in [13], where both the dealer and shareholders are not assumed to be honest. In that paper, the authors extend the term verifiable secret sharing to verify the shares, distributed by a dealer as well as shares submitted by shareholders for the secret reconstruction, and to verify the reconstructed secret.

   There exists many secret sharing schemes, but, due to Tor's limitations (low-latency, privacy) we will introduce Shamir's scheme in Tor as follows:

   We will divide the secret using Shamir's method.

   We will verify the correctness of shares.

   We will send shares to shareholder.

   In order to recover the secret:

   we will verify the correctness of pseudo-shares submitted,

   and we will reconstruct the secret using Shamir's method (Lagrange interpolation).

This paper is organized as follows. In the following section, we will briefly present Tor; in section 3, we will present Tor with a revocable anonymity. In section 4, we will propose some improvements in the use of Tor and before the conclusion; we will study the security of our design.

2 Tor

2.1 Presentation of Tor

The Tor network is composed by three main entities: directory servers, relays, and customers.
Directory server: it is assumed trustworthy, it manages the registration of new Tor nodes, and records for each Onion Router (OR), that is to say, its IP address, its Port onion, its public keys etc.
Tor client: it is the person or entity using the Tor network.
Relays: it is often composed by three nodes (by default) entry, middle, and exit, chosen randomly by the Tor client. Every relay contacts the directory servers to register itself as a relay and upload its key and configuration, such as open ports, and the advertised bandwidth.
As described in [6], Tor works as follows: Let's suppose that Alice is a Tor client. First, she queries a global directory to discover where on the Internet all the Tor servers are, in order to create a confidential network pathway with Tor. Then, Alice constructs its circuit incrementally, negotiating a symmetric key with each Onion Router on the circuit, one hop at a time. The authentication used is a first half of the Diffie-Hellman handshake, and it is done between Alice and each OR. No individual server ever knows the complete path that a data packet has taken.

2.2 Some weakness of Tor

There are several attacks which are possible in Tor [9], for example, If an adversary can see both ends of a Tor circuit, it can trivially correlate who is talking to whom over that circuit. This is generally known as an end-to-end correlation attack.
The attacker can also corrupt the traffic passing through a relay and then watch for corrupted traffic elsewhere in the circuit. This is sometimes called bit stomping or a tagging attack.
Even if it is claimed in [8] that: "only the Tor software on your computer knows about all three relays". We can admit that, the client will be able to get enough information on Tor bridges when it continues to download information about all relays in the network or most of existing relays and install malwares where it is possible.
The scalability of Tor is also a great concern, Tor is used throughout the world, but router participation is practically limited to only a few countries [12], which is inacceptable after the mass surveillance revelation.
As stated in [9] "... the current Tor network is not by itself sufficient to protect against all of the significant adversaries that may oppose law enforcement or defenders of national security. ... nothing prevents a large-scale adversary from breaking into or contributing many relays in many locations. An adversary that controls or observes a significant fraction of the network will observe a significant fraction of the network traffic. ... For correlation vulnerability, the communication links between the relays matter as much as the relays themselves, if not more. An adversary that can observe even a small number of strategic Internet links can do end-to-end correlation on a large fraction of the traffic that passes over Tor. Fortunately Tor is good enough for many of us. An abusive ex-spouse may be determined enough to monitor or break

into a mutual friend's communication to try to find you, but is still unlikely to be able to track you if you communicate with that friend over Tor. . . . Things are a bit more difficult, however, for those who oppose more resourceful and well resourced adversaries."
Our comprehension of the previous statement (made by one of Tor's designers) is that, Tor may protect against a small hacker but it may be a sucker trap for a powerful government.
We believe that the previous weakness is an important fact, that we should take into account, since it would be better not to have a Tor network, if this latter is weak. The reason is, this weakness endangers lives of thousands of its users.
It may be noted that if Tor has been unclassified [8], unlike its predecessor "onion routing" which were patented, it is perhaps due to the fact that it does not present a risk for organizations that have a huge potential in traffic analysis. However, the biggest advantage of this non classification is elsewhere, because if Tor had been classified, it would have been used by one group (military and diplomats of a same country), and in that case the traffic analysis would be easier. While the non classification allows them to have a large number of users which permits to conceal the identity of military and diplomats in those of journalists, cyber dissidents and drug traffickers.
We can note that the competitors of Tor are more vulnerable because of their small number of users [9].
To conclude this non-exhaustive list of the weaknesses, we can mention hidden nodes (bridges). Thinking that we can hide Tor bridges (which can be distributed by email) looks like trying to hide a cryptographic algorithm, which is at odds with the Kerckhoffs's principle. A5/2 encryption algorithm was designed for developing countries, due to the fact that A5/1 used in European countries, was too strong to export to the Middle East. Unfortunately for its designers, the description of A5/1 kept secret, was reversed engineered by Briceno et al. [10]. We cannot assimilate the concealment of bridges to the hiding of a private key, because private keys are never sent to a third party as it is the case here with bridges.

3 A revocable anonymity in TOR

As stated in the introduction, the revocation of the anonymity must obey a certain number of properties. We propose the following five properties, that any scheme using the revocation of anonymity on anonymous networks should respect.

1. The revocation of the anonymity must be a consensus among users of the network.
2. Individuals who have the ability to proceed to the revocation of the anonymity must be protected against potential threats or reprisals from the dishonest user.
3. Individuals who have accomplished the revocation and those who have refused must remain anonymous.
4. The introduction of the anonymity revocation must not weaken the security of the network.
5. The possibility of the anonymity revocation requires to ensure the authentication of the sender.

3.1 Design

Definition

- We define H as a secure collision resistant one-way hash function, which takes as argument a string of arbitrary length (secret part) and produces as output a binary string of a fixed length.
- The secret is defined here, as all informations which can help to identify the sender (IP address, ...)
- For a threshold secret sharing scheme (t, n), n is the number of shares and t the threshold (less than t shares cannot recover the secret).

Remark

1. In our scheme, we introduce the following modification in the functionality of Tor. We consider that, all Tor participants can be considered as Tor relays, that is to say, in our scheme, Tor clients are also relaying messages. If a Tor router receive a message from a Tor client, it can not known if the client is the sender, a bridge or a relay. To achieve this modification, we can add an option during the registration, in order to find those who are willing to relay sometimes or we can also oblige all Tor clients to relay sometimes (less than 10% of their activities).
2. The circuit is still taken randomly but by the directory server and not by the client.

Preliminary Steps of Alice:

- Alice will divide its secret (IP address, ...) into n parts, $p_1, \ldots, p_n$ and applies H on each part $(H(P_1), \ldots, H(P_n))$.
- Alice will hash the message m, which she wishes to send to Bob $(H(m))$.
- Alice will write in two different files $file_1$ and $file_2$, results of the hash function $(H(P_1),\ldots, H(P_n)$, $H(m))$.
- She will timestamp and sign $file_1$, which become now $file_{1signed}$ and she will send $file_{1signed}$ and $file_2$ to the directory server.

Preliminary Steps of the directory server

- From the knowledge of Alice's secret (IP address, the prime number chosen in the shares computation, etc.), the directory server will compute the secrets shares, verify the hashes, the timestamp and Alice's signature.
- The directory server will put in a secure storage the $file_{1signed}$ and it will take randomly three public keys and generate for any public key a signed roadmap where it is indicated the previous relay, the next relay, and the time when the message is supposed to arrive. For the last relay, only the previous relay and the time when the message is supposed to arrive is indicated on its roadmap.
- The directory server writes in each roadmap, the number of secret shares, which Alice would have to give to each node.
- The directory timestamps and signs $file_2$, which become $file_{2signed}$.
- The directory sends $file_{2signed}$ and all roadmaps to Alice, it will also delete all files, and shares, only $file_{1signed}$ is kept in a secure storage.

Step 1:
With the three public keys received from the directory server, Alice will encrypt the message as it is done currently with Tor, but taking care to add for each node, the roadmap, the secret shares (if any) and $file_{2signed}$, following the order indicated by the roadmaps.
The distribution of shares (Alice's IP address) to different nodes will observe the following rules:

- For a secret shared into n parts, a node can have between 0 to $t-1$ shares
- In order to hide Alice's identity to the entry node, the directory can allocate some shares to Alice, while ensuring that the shares held by other relays are sufficient to reconstruct the secret.
- Two different nodes cannot have the same secret share.
- Prior to deliver Alice's message to the recipient, it should be ensured that all parts of the secret have been distributed to the nodes.

If Alice receives secret shares attributed by the directory, she will write and sign in a new file $file_{3signed}$ the hash of the shares obtained, if not she will only create the file $file_3$.
As it is done currently, Alice will send its message to the entry node.
Step 2:

- The entry node (node1) will open the message with its private key, and it will ensure that the roadmap is followed, that is to say, the message comes from the Alice's IP and should go to the middle node.
- If node 1 does not have some secret shares, it forwards the packet to the next node.
- Otherwise, if node1 has some secrets shares, it will hash secrets shares and ensure that they match with those on $file_{2signed}$.
- If it does not, node1 will stop the process.
- If it matches, it will write in file $file_3$ the hash of shares which it received, and it will put in a secure storage its secret shares.
- It will sign the file $file_3$ which becomes $file_{3signed}$ and send it, accompagnied by the rest of the message encrypted by Alice to the middle node.

Step 3:

- The middle node will follow the same method (items of step 2).

Step 4:

- The exit node (Node 3) will open the message, verify that the roadmap is respected, check that all hash are cited in $file_{3signed}$ or the rest corresponds to the hash of secret shares that are in its message.
- It will also verify that, the hash of the message $(H(m))$, that Alice is trying to send anonymously to Bob, matches with the hash which is marked on $file_{2signed}$.

- If all verifications are conclusive, it will keep its shares and $file_{2signed}$ in a secure location and then send the message m to Bob.
- If there is a problem on the secret shares or the hash of the message, it will stop the process.

Remark:

- For each Tor relay, our modification adds on average, in computation time: the time required to use a hash function, to verify the signature of the directory ($file_{2signed}$), to verify the signature of the previous node ($file_{3signed}$) and to sign a file ($file_{3signed}$).

  According to [7], we can admit that RSA 1024 Signature will cost 1.48 ms and RSA 1024 Verification will cost 0.07ms, hence, our modifications on each Tor relay will cost approximately 2ms.

  All the cryptographic algorithms were coded in C++, compiled with Microsoft Visual C++ 2005 SP1, and ran on an Intel Core 2 1.83 GHz processor under Windows Vista in 32-bit mode.
- It is impossible to keep shares all the time in a secure storage, hence it is necessary to regularly delete files and shares that have existed for a while.
- When the communication is established with Bob, there is no more files ($file_{2signed}$, $file_{3signed}$, roadmap, ...) to send. In other words we did not change other parts of Tor.

3.2 The revocation of the anonymity

Step1:
The exit node will be contacted by a serious organization telling it that, the message sent before was a ransom demand.
The last node will check if the hash of that message matches with the hash present in $file_{2signed}$ ($H(m)$), and it will also verify the validity of the timestamp.
Step 2:
If the previous verifications are conclusive, the last node will publish the motivated request and the file $file_{2signed}$.
Step 3:
As a winning lotto ticket, it will appear somewhere on the web browser, the hashes corresponding to secret shares.
Step 4:
The supposed offender must be able to defend itself (telling its opinion).
Step 5: Users of the Tor circuit will try to reconstruct the secret, in order to decide between the two parties. Those who have decided to break the anonymity of the alleged offender will send their shares, if the organisation receives at least t shares among n for a threshold scheme (t, n), then it can reconstruct the secret.
If the organisation receives less than t shares, this means that network members have refused to disclose the identity of the sender. The directory server could also send $file_{1signed}$ which can be legal evidence of the involvement of Alice.
Remark

- In practice, the fact that most of the nodes are in the same geopolitical space [12], could facilitate the removal of certain anonymity and the refusal to revoke others.
- Shares must be sent anonymously.
- Nodes which received shares during the process and which are not connected during the demand for the reconstitution can be considered as abstainers.

3.3 Remarks

In our scheme, we introduced the following modifications in order to achieve the five rules necessary for the revocation of the anonymity.

1. To achieve a consensus (rule 1), we found useful to introduce a verifiable secret sharing scheme using the Shamir's scheme.
2. In order to protect those who have the ability to revoke the anonymity against potential threats (rule 2), we found useful to reduce the information which the sender has on relays. Thus, the directory server can set up a group encryption where multiple relays would have the same public key. In the case where some group encryption [11] may be difficult to implement in Tor (low-latency, privacy, etc.), we could try to work around, like binding a public key to several IP ...
3. Those who have executed the removal of anonymity will remain anonymous (rule 3), because they would send the shares allowing the reconstruction of the secret anonymously.
4. We must recognize that the elimination of unexpected exits, or the introduction of the time in roadmaps could perhaps reduce the capacity of some users.
5. Signature, timestamp and hash will ensure the identity of Alice in due time (rule 5).

3.4 A revocable anonymity in hidden services

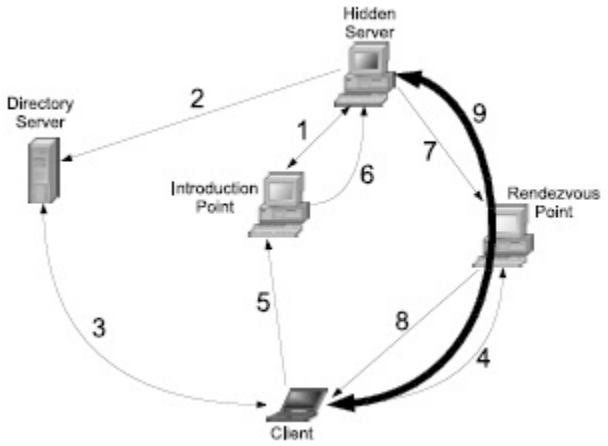

Figure 1: Normal use of hidden services and rendezvous servers.

As described in [5], a normal setup of communication between a client and a hidden service is done as shown in *Figure1*.
First the Hidden Server (HS) connects (1) to a node in the Tor network and asks if it is OK for the node to act as an Introduction Point for his service. If the node accepts, we keep the circuit open and continue. Next, the Hidden Server contacts (2) the Directory Server (DS) and asks it to publish the contact information of its hidden service. In order to retrieve data from the service the Client connects (3) to DS and asks for the contact information of the identified service and retrieves it if it exists (including the addresses of Introduction Points).
In our scheme we keep all the circuits of *Figure 1* as such, except for two connections ((2), (3)) concerning the directory server that we modify as follows:
(2): When the Hidden Server contacts (2) the Directory Server (DS) and asks it to publish the contact information of its hidden service. As it is done in section 3.1, HS will also send a file $file_{1signed}$ where it has written and signed the hash of its secret shares (Hidden server's IP address shared).
The DS will recompute and verify the validity of hashes and signatures present in $file_{1signed}$, if verifications are conclusive, then it will keep in a secure storage secret shares and $file_{1signed}$.
(3): When the Client connects to DS asking for the contact information of the identified service (including the addresses of Introduction Points).
The DS will create $file_{2signed}$ (as it is done in section 3.1) and send the answer and $file_{2signed}$ to the client, via a circuit chosen randomly, while sending the answer, it will also send secret shares to members of that circuit (3).
At the end of that process, the directory server will delete all the secret shares (concerning this hidden server) which was in its storage.
When another client connects to DS asking for the contact information, the DS will send it the answer and $file_{2signed}$ (no secret shares).
Revocation of the anonymity
Let's suppose that the client has a proof showing that this Hidden server is promoting child abuse, the client will exhibit to all Tor users $file_{2signed}$ and its proof.
Members of that circuit (3) will send their shares if they are convinced, otherwise they will not.
Remark
There is no roadmap, timestamp and $file_{3signed}$ in this scheme, due to the fact that the DS is trustable in the distribution of the shares which is not the case of Alice in section 3.1.

4 Other possible improvements

There are in the literature many papers focusing on improving Tor's latency, here, the proposed improvements are oriented on the anonymity of Tor users.

4.1 Mixing with a lot of data

Let's suppose that groups like Cisco, Google, Facebook, Brazil telecom, Baidu, Akamai ... are very unhappy with the mass surveillance and they have decided to participate in the common effort to reduce mass surveillance, by mixing their data with data coming from Tor users.
In order to see how the mixture of data can improve the anonymity, let's take a concrete example. A person is prosecuted in the street by opponents who are armed and are at least 1000 times faster than him.

We believe that, one of his single chance of survival is to move very quickly in a dense crowd of people, who looks like him as two drops of water. Transposed to the case of the Internet, it would be to mix Tor's packets to billions of others which are online at this moment.
It remains to protect packets against attacks such as fingerprint [14], traffic injection, etc.
Let's suppose that an agency which records all communications that are in the world decides to discover the name of Alice's contacts:

1. The agency would see that Alice sent a packet X at time t to the first Tor relay.
2. By analyzing the communication of the first Tor relay, the agency would know that it has sent a packet at time t to Google.
3. By analyzing packets sent by Google at that time t, the agency will find that hundreds of millions or billions of packets are sent in all directions.

The agency will have to find the right packet X among the billions others, it should also be able to convince a jury that the recipient of the package is the contact of Alice. While in the current system, if an agency records all communications that flow through the Tor network, it can discover the name of the potential contact of Alice quickly.
It is possible that some users will be reluctant to use big internet groups (Google, Yahoo, Facebook, Wikipedia, Akamai ....) to protect their anonymity, we just tell them this: These are the only places on the internet where there is a dense crowd of data, in a such way that they can get rid of your footsteps.
In this case, Tor can be built in such manner that the big Internet groups will carry packets without knowing their contained, their origin and their destination.
In summary, the security of Tor should not only depend on the hidden nodes, but on tens of billions of packets that are processed together in a network (network mix) and hundreds of millions of existing relays (some of which may be used for a few seconds).

4.2 Camouflage

It is well known in the literature [14], that the padding and the camouflage are valid methods of protection against traffic analysis.
As proposed in [14], the use of camouflage which is a smarter way to intentionally change the patterns of the data traffic can be interesting in Tor. For example, it would be interesting to consider adding random time during the passage of packets in the Tor nodes, without compromising its low-latency aspect.
It could also be interesting to trigger (with a pseudo random generator) random packets which would be sent to random destinations.
Using these random packets will largely render obsolete the interest of traffic analysis, since it will be difficult to know who is talking to who.

4.3 Honeypot

A Tagging attack can be defined as a hostile node tagging a cell by altering it. We would suggest a kind of honeypot for such attacks. if a node detects a tagging attack on a packet X, it formats the packet X in the normal format (cell), after that it creates a real fake packet in which the node will add the tag detected and send it on a destination chosen randomly. While the packet X will follow its normal path as expected.

4.4 Allow users to choose between safety and speed

Tor users should have the possibility to choose between increased anonymity and low latency increased.

4.5 Accessibility of Tor software

Tor should be easier to acquire, for example, it would be interesting to have Tor in all web browsers or all Operating System.

5 Security analysis

In addition to attacks inherent to Tor listed in several papers [9], we also have three types of attacks, which could arise from changes we have introduced, such as the case of a user who is trying to cheat to remain anonymous in any case. We also have the case of those who try to impersonate some Tor users in order to commit crimes with their identities. Finally, we have attackers who may try to revoke the anonymity of a Tor user, without the consensus of the majority of circuit members.

5.1 Attack of the user who wishes to remain anonymous in all cases

The attacker refuses to distribute some shares
If Alice tries to hide her IP address by distributing a number of shares which does not allow the reconstruction of the secret (for example, in a threshold scheme (t, n), Alice can distribute less than t shares). In this case, it is in the responsibility of the last node to reject the Alice's request.
Attack with help of an accomplice
It should be noted that the help of an accomplice present in the random circuit can allow Alice to perform an unexpected exit, instead of the normal roadmap which was assigned to her. We recall that the presence of an accomplice in a random circuit is highly unlikely in a practical case.
Distribution of corrupted shares
If Alice decides to prevent the reconstruction of its IP address by distributing false shares, she would be detected quickly, since $file_{2signed}$ contains the hash of all shares.
Injecting a corrupted file $file_{3signed}$
Let's suppose that Alice has created a file $file_{3signed}$ where she writes the hash of all secret shares which are in $file_{2signed}$ except for shares that are intended for the last node.
After that, she sends to the last relay, its shares, $file_{3signed}$ and $file_{2signed}$.
This attack will also be detected, since the roadmap indicates that Alice is not the penultimate relay and $file_{3signed}$ must be signed by the penultimate relay (as indicated in the roadmap).
Sending corrupted shares during the reconstruction of the secret
The file $file_{2signed}$ is sufficient to enable one who reconstructs the secret to separate the good parts from the false shares.
Threaten members of the circuit to prevent the reconstitution of the IP address.
Alice should not be able to identify the different relays of its packet, because it will be a group encryption or something similar which would be used.

5.2 Revocation of the anonymity of the user without the consensus of the majority of the circuit.

An attacker can try to revoke the anonymity of Tor users without the consent of the Tor circuit members.

A node may try to find its position in the circuit

This is impossible, since a node can receive between $t-1$ shares and 0 shares, thus having the file $file_{3signed}$ empty does not mean that you are the first node, and having some hash in $file_{3signed}$ does not mean that you are the second node. However, the last node knows its position.

An attacker could guess the circuit by looking into Onion routers (OR)

This attack is difficult to achieve, since it would need a search in a big number of OR (in our scheme, Tor users are also OR), we recall that OR must delete all logs and files, only the secret (and $file_{2signed}$ for the last node) should be stored securely .

We should note that, while in classical Tor an attacker needs to find all members of the circuit, to revoke the anonymity of the user, in our model, it needs less OR since finding t secret shares would be sufficient to revoke the anonymity. One solution would be to set up a (n, n) secret sharing scheme , which we did not want to implement, because we prefer the removal of anonymity on the basis of a majority, instead of revoking the anonymity on the agreement of all members of the circuit.

Attack on the roadmap

If none of relays has respected the instruction consisting on deleting roadmaps, then it is possible that the attacker can reconstruct the circuit and thus endanger the anonymity of the user.

Attack on secret shares

In case of attack, a "legal" action or illegal coercion, a node cannot revoke the anonymity alone. Similarly, if the directory server complied with rules of the game (delete all elements of logs and keep only $file_{1signed}$) then it is impossible to revoke the anonymity without the participation of those who received shares of the secret.

End-to-end timing correlation

Could the timestamp which we have introduced, help in the End-to-end timing correlation? Partly but not necessarily, since the file is not timestamped for the outputting circuit, and between the last relay and the true destination of the packet (Bob) there is no trace of timestamp.

5.3 Impersonate Tor users

The revocation of the anonymity could encourage malicious people, to try to impersonate Tor users using the following methods.

IP spoofing

Let's suppose that by illegal means, Trudy could spoof the IP address of Alice, but he can not pretend to be Alice if he does not have Alice's private key.

Spoofing $file_{2signed}$

The attacker cannot use $file_{2signed}$ effectively in order to accuse Alice, if it does not know Alice's IP address. If the attacker knows Alice's IP, then timestamps, hashes and shares (it can not invert the hash to find shares) should prevent a dishonest use.

Spoofing $file_{3signed}$

An attacker might be tempted to steal $file_{3signed}$, to modify it and inject it back into the circuit. However, no changes are possible on this file, since it is modified and signed by each node, and the roadmap has to be followed.

Spoofing roadmaps
It may be noted that there is no relationship between a roadmap and Alice, then if Trudy manages to spoof roadmaps (in the time interval required for their use), it would bring him anything because it is assumed that the circuits have been selected randomly.
Other attacks
A malicious node could delay Alice's message in order to create a stop delivery (due to the timestamp). It could also create false files ($file_{2signed}$, $file_{3signed}$, roadmaps) to stop deliveries but this type of attack also exists in the current Tor (an attacker can block the packet sent).
Attack on the directory server
Our configuration creates more risks to the directory server, if it breaks, all the circuit is at risk (roadmaps, files, etc).
To prevent this attack, the directory server should be more protected and Alice should not be on the same geopolitical territory to prevent a "legal" coercion.
However, we recall that, even in the classical Tor, a malicious directory server can send a list of malicious nodes to Tor clients.
If the directory server respects the rules consisting in the delete of all files concerning Alice except $file_{1signed}$, then there is no possibility for the attacker to guess the Alice's identity with that file ($file_{1signed}$).
To prevent a denial of service on the directory server, it would be useful to increase the number of the DS, since they would be in great demand with the configuration that we have proposed. It is also possible to prepare the communication long time before sending the message, to reduce the time intended to send the message.

6 Conclusion

In that proposed design we tried to improve Tor with a little change and we also tried to discourage drug traffickers, spies, and pedophiles from using Tor.
In this paper, we always assume that the directory server is trustworthy, however, in a world where some organisations have tremendous opportunities, we should have in the future a deep reflection, on a system where even in case of a fall, the directory server would not lead in its fall the whole system.
We can hope that Tor is not another Enigma machine [15] (after the World War II, the Allies sold Enigma machines, to developing countries. As these countries did not know that the machine had been broken, their "secure" communications were being intercepted by the Western intelligence agencies), however, as it is done currently, it is certain that Tor favors intelligence services that have huge resources.

References

[1] B. Muir *TOR Packet Analysis - Locating Identifying Markers*,locating unique identifying markers, for InfoSec management (LE/ISP MiTM visibility). 2010.

[2] *Japanese Police target users of Tor anonymous network*

http://www.bbc.com/news/technology-22248692

[3] M. Wright, M. Adler, BN. Levine, C. Shields *An analysis of the degradation of anonymous protocols*, 2002.

[4] Shamir, A. *How to share a secret*.Common.ACM 22 (11) (1979) 612-613.

[5] Overlier L., Syverson P. *Locating hidden servers.* In Security and Privacy, 2006 IEEE Symposium on (pp. 15-pp).

[6] ssd, *The Surveillance Self-Defense* https://ssd.eff.org/tech/tor

[7] Cryptopp, *http://www.cryptopp.com/benchmarks.html*.

[8] Dingledine R., Mathewson N., Syverson P. *Tor: The second-generation onion router.* Naval Research Lab Washington DC, (2004).

[9] Syverson P. *Practical Vulnerabilities of the Tor Anonymity Network.* Advances in Cyber Security: Technology, Operation, and Experiences, (2013).

[10] Briceno M., Goldberg I., Wagner D. *A pedagogical implementation of A5/1*. (1999).

[11] Kiayias A., Tsiounis Y., Yung M. *Group encryption.* In Advances in Cryptology-ASIACRYPT 2007 (pp. 181-199).

[12] McCoy D., Bauer K., Grunwald D., Kohno T., Sicker D. *Shining light in dark places: Understanding the Tor network.* In Privacy Enhancing Technologies (2008, January), (pp. 63-76).

[13] Parmar K., Jinwala D. *A Novel Approach for Verifiable Secret Sharing by using a One Way Hash Function.* arXiv preprint arXiv:1203.3620.

[14] Panchenko A., Niessen L., Zinnen A., and Engel T. *Website fingerprinting in onion routing based anonymization networks.* WPES, page 103-114. ACM, 2011.

[15] Winterbotham, F.W, *The Ultra Secret*, Harper and Row, New York, 1974; Spanish edition Ultrasecreto, Ediciones Grijalbo, Madrid, 1975.

[16] Wikipedia, http://en.wikipedia.org/wiki/Shamir's_Secret_Sharing

7 Example

Definition

- We denote by E the encryption function using the key X and the plaintext Y, hence we denote the cipher obtained with this function by $E(X : Y)$.
- We denote by $Pub_{Keyuser1}$, the public key of user1.
- We denote by || a separation between two parts of a message.

Preliminary Steps of Alice:
In order to illustrate our scheme, let's take an example of secret sharing scheme given by Wikipedia[16]. In order to simplify the explanation, the calculations in the example are done using integer arithmetic rather than using finite field arithmetic.
Let's suppose that Alice's IP address is 1234. Alice uses the Shamir Secret sharing scheme in order to divide its IP into 6 parts $(n = 6)$, where any subset of 3 parts $(t = 3)$ is sufficient to reconstruct the secret.
Alice takes randomly two $(t - 1)$ numbers: $(a_1 = 166, a_2 = 94)$.
Alice's polynomial to produce secret shares (points) is therefore: $f(x) = 1234 + 166x + 94x^2$.
Alice constructs 6 points $D_{x-1} = (x, f(x))$ from the polynomial: $D_0 = (1, 1494)$; $D_1 = (2, 1942)$; $D_2 = (3, 2578)$; $D_3 = (4, 3402)$; $D_4 = (5, 4414)$; $D_5 = (6, 5614)$.
Alice hashes each part of the secret with SHA1 (SHA1 is taken just as an example; in a real case she may take SHA2).
$HD_0 = SHA1((1, 1494)) = 9D13867F7AC77578AD166DFD4903929C94C1D345$.
$HD_1 = SHA1((2, 1942)) = FE2EC1340A7F69BECE97EC8D7BC6FBA6F7D15266$.
$HD_2 = SHA1((3, 2578)) = C9E713E100E82E3217B4078CFE76F1FA04504E62$.
$HD_3 = SHA1((4, 3402)) = 95226F5EAAF1954B14561C8B3720F3D52ABCD3C4$.
$HD_4 = SHA1((5, 4414)) = 003D3D1107B93A6F64B40145B55BE107616D8FFE$.
$HD_5 = SHA1((6, 5614)) = 51819611ADCE4041767D1412D6C215343563D5D1$.
She also hashes the message which she wishes to send to Bob (we suppose that message is: "Hi, I am Alice").
$HM = SHA1(Hi, I\,am\,Alice) = C70B95E2234CF8B7FE67F7B11F1DC07F31303476$.
She creates two file $file_1$ and $file_2$ where she writes $HM, HD_0, HD_1, HD_2, HD_3, HD_4, HD_5$ and the time of request. She signs $file_1$ which becomes $file_{1signed}$ and sends to the directory server $file_2$, $file_{1signed}$, a_1 and a_2.
Preliminary Steps of the directory server.
From Alice's IP address, a_1 and a_2, the directory reconstructs Alice's polynomial $f(x) = 1234 + 166x + 94x^2$ and verifies validities of hash and time. If all verifications are conclusive, then it signs $file_2$ which becomes $file_{2signed}$ and it takes at random 3 public keys (Rougon, Macquart and Zola).
The directory issues a signed roadmap for each participant (Rougon, Macquart and Zola). For example in $Roadmap_{Rougon}$ we can see:

IP address of sender: 1234 (Alice)
IP address of receiver: 2345 (Macquart)
Number of secret shares assigned: 2

Arrival time: time1.
Timestamp and signature of directory: Signed by directory server on time1

We will suppose, in the rest of the example, that the directory has assigned two secret shares to Macquart, one to Zola, two to Rougon and one to Alice. The directory sends securely, $Roadmap_{Rougon}$, $Roadmap_{Macquart}$, $Roadmap_{Zola}$, and $file_{2signed}$ to Alice. The directory server puts in a secure location $file_{1signed}$ and deletes all other informations concerning Alice.

Step 1:

Alice finds on $Roadmap_{Zola}$, that the directory has assigned one secret share to Zola, then Alice encrypts with the public key of Zola, the following elements: 1 secret shares (D_2), the roadmap for Zola, the $file_{2signed}$ (containing the hash and timestamp) and the message m for Bob.

$$C_1 = E(Pub_{KeyZola} : (3, 2578)||Roadmap_{Zola}||file_{2signed}||m).$$

After that, Alice encrypts with the Macquart's public key the following elements: 2 shares of the secret (D_4, D_5), the roadmap for Macquart $(Roadmap_{Macquart})$, the file $file_{2signed}$ and C_1 the message for Zola.

$$C_2 = E(Pub_{KeyMacquart} : (5, 4414)(6, 5614)||Roadmap_{Macquart}||file_{2signed}||C_1).$$

Note that, to hide the number of remaining relays, the directory has assigned one secret share to Alice, that is why, Alice will create and write in $file_{3signed}$ the hash value of its share (HD_0).
Alice then encrypts with the public key of Rougon, 2 shares (D_1, D_3), the roadmap for Rougon, $file_{2signed}$, $file_{3signed}$, and C_2 the package for Macquart.

$$C_3 = E(Pub_{KeyRougon} : (2, 1942)(4.3402)||Roadmap_{Rougon}||file_{2signed}||file_{3signed}||C_2).$$

Alice sends C_3 to the first relay.

Step 2:

Rougon will decrypt with his private key, to find the encrypted message for Macquart, his roadmap, two shares ((2, 1942) (4.3402)), $file_{3signed}$, and the file $file_{2signed}$.
Rougon will consult his roadmap signed by the directory server, which will allow him to know that at this moment, he should receive a message from Alice going to Macquart.
Rougon will apply the SHA1 hash function on the two secret shares $(SHA1((2, 1942))$ and $SHA1((4.3402)))$ to verify the presence of HD_1 and HD_3 in $file_{2signed}$.
If this verification is successful, Rougon will keep in a secure location the two shares ((2, 1942) and (4.3402)). He will add in a new file $file_3$ the hash HD_1 and HD_3 in addition to HD_0. Rougon will sign $file_3$ which becomes $file_{3signed}$.
Rougon will send to Macquart $E(Pub_{KeyMacquart} : file_{3signed}||C_2)$.
Otherwise, if HD_1 and HD_3 are not present in $file_{2signed}$, or if HD_1 and HD_3 are already present in $file_{3signed}$, then Rougon will stop the process.

Step 3:

Macquart will open the message with his private key, to discover $file_{3signed}$, the encrypted message for Zola, his shares, his roadmap and $file_{2signed}$. He will apply $SHA1$ on his shares and verify if HD_4 and HD_5 are in $file_{2signed}$. He will also check the validity of his roadmap.
If this verification is successful, Macquart will keep in a secure location the two shares of the secret (D_4, D_5). He will add in a new file $file_3$, HD_4 and HD_5 in addition to HD_0, HD_1 and HD_3. Macquart will sign $file_3$ which becomes $file_{3signed}$.

Macquart will send to Zola $E(Pub_{KeyZola} : file_{3signed}||C_1)$.
Otherwise, if HD_4 and HD_5 are not present in $file_{2signed}$, or if HD_4 and HD_5 are already present in $file_{3signed}$, then Macquart will stop the process.

Step 4:

Zola will open the message with his private key, in order to find $file_{3signed}$, the encrypted message for Bob, his share of the secret, his roadmap and $file_{2signed}$.
He will apply SHA1 on his share to verify if HD_2 is present in $file_{2signed}$. He will also check if HD_0, HD_1, HD_3, HD_4 and HD_5 are already in $file_{3signed}$.
After that he will check in $file_{2signed}$ if $SHA1(m)$ matches with the hash of the message that Alice is trying to send to Bob.
He will also check the validity of his roadmap.
If all verifications are conclusive, he will send m to Bob and he will keep in a secure storage D_2 and $file_{2signed}$.
Otherwise, if HD_2 or SHA1(m) are not present in $file_{2signed}$, or if HD_2 is already present in $file_{3signed}$, then Zola will stop the process.

The revocation of the anonymity

1. Zola receives a message from a serious organization requesting for the revocation of the anonymity of the user who sent the message m_1.

2. Zola will check in the multitude of $file_{2signed}$ which he has recorded, the file where $SHA1(m)$ is equal to $SHA1(m_1)$.

3. Zola will broadcast to all users the anonymity revocation request and hashes present in $file_{2signed}$ (i. e. HD_0, HD_1, HD_2, HD_3, HD_4, HD_5).

4. Tor users will study the request of that organization, those convinced by this demand will send their shares.

5. We assume that Rougon is not convinced by the arguments of the organization, Macquart and Zola will send $(5, 4414)$, $(6, 5614)$ and $(3, 2578)$.

6. The organization must ensure that the hash of the shares received are present in $file_{2signed}$ before reconstituting Alice's IP address using the Lagrange polynomial.