

# A revocable anonymity in Tor

Amadou Moctar Kane

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

March 6, 2015

## Abstract

This new protocol is based on the idea of introducing a revocable anonymity in Tor, which was presented in our recent paper entitled "Another Tor is possible". Compared to that previous paper, this present scheme simplify the first protocol and reduce the power of the directory server, while maintaining the ability for the Tor community, to break the anonymity of a sender in case of misconduct.

We also take the opportunity of this paper, to appeal the majors internet companies, to help in the creation of a responsible Tor network (without pedophiles, spies, ....), by mixing billions of data flowing through their networks with those of Tor.

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

## 1 Introduction

Tor (The Onion Router) is a free software designed for online anonymity and resisting censorship. In the NSA top-secret documents revealed by Edward Snowden, it is described as "The king of high-secure, low-latency anonymity". Unfortunately Tor is now the haunt of pedophiles, spies and other criminals. In "Another Tor is possible" [9], we introduced in Tor a system that will ensure anonymity for all its 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 our previous scheme [9], we supposed that the directory server was trustworthy, however, in the era of mass surveillance, we may also think about a curious directory server (DS). In that first scheme, the curious (DS) could guess the correspondent of Alice, by monitoring the exit node which it had chosen for her, while in the present scheme, the DS is no longer involved in the choice of Alice's nodes, since she is free to choose relays she want.

In our point of view, 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.

Having a responsible Tor (without criminals) would put big internet companies face their responsibilities ie ensuring that their users have responsible anonymity when using their applications (protect the users of their networks as long as those ones remain within the limits of legality).

On the other hand, the current Tor (with criminals) will always be an excuse for governments and big Internet companies to refuse to participate in the fight against mass surveillance. But beyond these pretexts what would we do if our child was kidnapped by a pedophile using Tor? would we be willing to help Tor after that? would we try to break the anonymity of the offender?

In addition, can we oblige someone who is trying to help cyber dissidents to also help pedophiles, drug traffickers and the NSA while it is against its philosophy?

Therefore, we should give to all the Tor relays the possibility to say: I want protect this activity and I do not want to protect this other and this improvement is possible in our two schemes.

If things remain as such, it is likely that these companies could one day be subject to prosecution for negligence or complicity. The smaller support that big internet companies could bring to Tor, would be to mix the data of Tor with theirs. In order to see how the mixture of data can improve the anonymity, let's take a concrete example. Someone is prosecuted in the street by opponents who are armed and are at least one thousand times faster than him. We believe that, one of his single opportunity 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.

In order to have an effective mixture, it is obvious that all data flowing through this network must be in the same format (encryption, size, ...).

## 1.1 Preliminaries

**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.

**Secret sharing schemes:** One of the two first secret sharing schemes was introduced by Adi Shamir [3]. 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 approach (definition) for verifiable secret sharing presented in [11], 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 share holders 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 illustrate our proposal with the Shamir's scheme 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).

## 1.2 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 [5], 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.

## 1.3 The revocable anonymity presented in our previous paper [9]

In brief, in that scheme, we introduced the following modifications in the functionality of Tor.

1. We consider that, all Tor participants can be considered as Tor relays.
2. The circuit was taken randomly by the directory server and not by the client.
3. From the knowledge of Alice's IP address, the directory server computes the secrets shares, and allows some shares to each member of the circuit.
4. The directory server takes randomly three public keys and generates for any member of the circuit a signed roadmap where it is indicated the previous relay, the next relay, the time when the message is supposed to arrive and the number of allocated secret shares.
5. 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, a file *file<sub>2signed</sub>* containing all the hash of the shares, and *file<sub>3signed</sub>* containing the hash already received by the previous relays following the order indicated by the roadmaps.
6. When a relay receives the message, it will verify if *file<sub>2signed</sub>* and *file<sub>3signed</sub>* are correct and if the roadmap is respected.
7. If all verifications are conclusive, the relay will keep its shares and *file<sub>2signed</sub>* in a secure location and then sends the message to the next relay.
8. If there is a problem on the secret shares or on the hash of the message, then it will stop the process.

This paper is organized as follows. In the following section, we will present our new scheme; and before the conclusion; we will study the security of our design.

## 2 The new scheme

The main contribution of this new scheme is to suppress the choice of the circuit which was done by the DS (return to the user the right to choose the circuit itself), to reduce the steps and to simplify the protocol. As stated in [9], the revocation of the anonymity must obey a number of properties. Our aim in this scheme, would be to respect the following five properties.

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 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 introduction of the anonymity revocation requires to ensure the authentication of the sender.

### 2.1 Design

#### Definition

- We define  $H$  as a secure collision resistant one-way hash function, which takes as argument a string of arbitrary length (the secret share) 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 new scheme, we consider that, all Tor participants can be considered as Tor relays, that is to say, in our scheme, Tor clients relay also messages. If a Tor router receives a message from a Tor client, it can not know 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 client and not by the directory server.

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.
- Two different nodes cannot have the same secret share.

#### *Preliminary Steps of Alice:*

- From the knowledge of the secret (Alice's IP address), Alice will compute the  $n$  secrets shares and writes in  $file_1$ , the hashes of the shares and the hash of the message intended to Bob ( $H(m)$ ).
- Alice will initiate an authentication with the directory server (DS) and send to the DS  $file_1$ .

**Preliminary Steps of the directory server**

- The DS will recompute and verify the correctness of the shares, after that it will timestamp and sign  $file_1$  which become  $file_{1signed}$ . It will also, send  $file_{1signed}$  to Alice.
- The directory server will send the secret shares to  $n$  members of Tor network chosen randomly.

**Step 1:**

Alice will encrypt the message as it is done currently with Tor, but taking care to add into the message of the last relay, the file  $file_{1signed}$ .

As it is done currently, Alice will send its message to the entry node.

**Step 2:**

- The entry node (first relay) will decrypt and send the message as it is currently done in Tor.

**Step 3:**

- The middle node (second relay) will decrypt and send the message as it is currently done in Tor.

**Step 4:**

- The exit node will decrypt the message, verify the validity of the timestamp and signature present in  $file_{1signed}$ . 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  $file_{1signed}$ .
- If all verifications are conclusive, it will keep  $file_{1signed}$  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 the first two relays, there is no modifications on Tor.
- For last Tor relay, our modification adds on average, in computation time: the time required to use a hash function, and to verify the signature of the directory server ( $file_{1signed}$ ).

According to [6], we can admit that RSA 1024 Signature will cost 1.48 ms and RSA 1024 Verification will cost 0.07ms, hence, our modifications on the last Tor relay will cost approximately 0.07ms.

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.
- Compared to our first scheme [9], we can notice that there is no roadmap, no  $file_{2signed}$  and no  $file_{3signed}$  in this protocol, due to the fact that the DS is trustable in the distribution of the shares which was not the case of Alice.

## 2.2 The revocation of the anonymity

### **Step 1:**

The exit node will be contacted by a serious organization asking for a revocation of the anonymity.

The last node will check if the hash of that offending message matches with the hash present in  $file_{1signed}$  (namely  $H(m)$ ), 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  $file_{1signed}$ .

### **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 via a hidden server).

**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 hash and compare their hashes with the hashes contained in  $file$ . If it matches, then they 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.

Otherwise, if the organisation receives less than  $t$  shares, this means that network members have refused to disclose the identity of the sender.

### **Remark**

- In practice, the fact that most of the nodes are in the same geopolitical space [10], could facilitate the removal of certain anonymity and the refusal to revoke others.
- Shares must be sent anonymously to the organization asking for the removal of the anonymity.
- Relays which received shares during the process and which are not connected during the demand for the reconstitution can be considered as abstainers.

## 2.3 Remarks

In this 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) on the removal of the anonymity, 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 of the offender (rule 2), we found useful to distribute the shares to  $n$  members of the network chosen randomly (Alice does not have an information on those who receive the shares).
3. Those who have removed the anonymity will remain anonymous (rule 3), because they would send the shares allowing the reconstruction of the secret anonymously.
4. It is certain that the introduction of an anonymity revocation will reduce the security of Tor ( $t$  members can break the anonymity of another member). However, the introduction of new security mechanisms (mixing data, camouflage, honeypot, ...) should compensate this reduction.

5. Authentication, signature, timestamp and hashes will ensure the identity of Alice in due time (rule 5).

## 2.4 A revocable anonymity in the hidden services

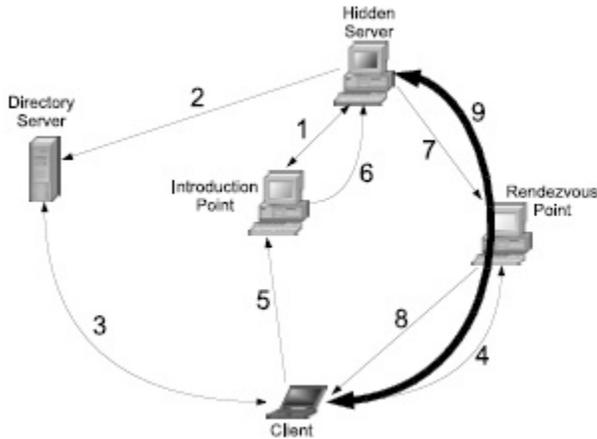


Figure 1: Normal use of hidden services and rendezvous servers.

As described in [4], a normal setup of communication between a client and a hidden service is done as shown in *Figure 1*.

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:

Connection (2): The Hidden Server contacts the Directory Server (DS) and asks it to publish its contact information. As it is done in section 2.1, HS will authenticate itself and the DS will generate a file  $file_{1signed}$  where it has written and signed the hash of the secret shares (Hidden server's IP address shared). It will send the secret shares to  $n$  members chosen randomly among the Tor members. After that, the directory server will delete all the secret shares (concerning this hidden server) which it had computed.

Connection (3): The Client connects to DS asking for the contact information of the identified service (including the addresses of Introduction Points).

In its answer, the DS will send  $file_{1signed}$  (as it is done in section 2.1) and the contact information to the client, via a circuit chosen randomly.

### **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 members  $file_{1signed}$  and its proof.

Members of Tor will send their shares if they are convinced, otherwise they will not.

### 3 Security analysis

In addition to attacks inherent to Tor and those listed in our first version [9], we also have some attacks, which could arise from changes we have introduced. We still have three cases, such as when a user 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 Tor members.

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

##### *The attacker refuses to send $file_{1signed}$ to the last node*

In this case, it is in the responsibility of the last node to reject the Alice's request. It may be noted that if the last relay tries to help Alice to cheat then its liability could be engaged, for example, if it transfers the message to Bob without having received a correct file  $file_{1signed}$ .

##### *Attack with an old version of $file_{1signed}$*

It should be noted that if Alice tries to send an old version of  $file_{1signed}$ , then the timestamp, the signature and the hash of her message should prevent this cheating.

##### *Threaten members of the network to prevent the reconstitution of the IP address*

Alice will not be able to identify the different members which have a share of her IP address.

#### 3.2 Revocation of the anonymity without the consensus

An attacker may try to revoke the anonymity of Tor users without the consent of Tor's members.

##### *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 should note that, in the classical Tor an attacker needs to find all members of the circuit, to revoke the anonymity of the user, while in our model, it needs to find  $t$  secret shares 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 our aim is to introduce a removal of the anonymity on the basis of a consensus, instead of revoking the anonymity on the agreement of all Tor's members.

##### *Attack on secret shares*

In case of attack, of 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 the shares after their distribution) then it is impossible to revoke the anonymity without the participation of those who received the shares of the secret.

##### *End-to-end timing correlation*

Could the timestamp which we have introduced, help in the End-to-end timing correlation? No it should not be the case, 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.

#### 3.3 Impersonate Tor users

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

### ***IP spoofing***

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

#### ***Spoofing $file_{1signed}$***

The attacker cannot use  $file_{1signed}$  effectively in order to accuse Alice, if it does not know Alice's IP address. If the attacker knows Alice's IP address, then timestamps, signatures and hashes (it cannot invert the hash to find Alice's message) should prevent a dishonest use. We also suppose that the communication between Alice and the DS is secure.

#### ***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 a fake  $file_{1signed}$ , 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***

If we compare with the first version [9], this configuration reduces the involvement of the directory server, however if it breaks, all the circuit is at risk (file, shares, 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.

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 the secret shares, then there is no possibility for the attacker to guess the Alice's identity with  $file_{1signed}$ .

To prevent a denial of service on the directory server, it would be useful to increase their number, since they would be in great demand with the configuration that we have proposed. It is also possible to prepare the communication a long time before sending the message, in order to reduce the time intended to send the message. We may also think about parallel computing and group signature for some Tor users which would play the role of directory server.

## **4 Conclusion**

Tor should not be secure for its sole users, but it should also be secure for its non-users (victims of pedophiles, of NSA, etc.) by introducing a revocable anonymity.

Here, we also tried to reduce the power allocated to the directory server in our first version, by returning to Tor's user the possibility to choose its circuit itself.

In addition, in this scheme we slightly modified Tor, since among the three existing relays, there is no change for the first two, only the third relay performs very slight operations in computation time.

Finally, in order to improve the security of Tor users, we also presented in this paper and in our first version [9], new types of protection such as mixing Tor data with big data, introducing a honeypot and a camouflage in Tor.

In the current Tor and in this paper, the directory server is supposed trustworthy, however, in a world where some organisations have tremendous opportunities, we should have in the future a deep reflection, on a peer-to-peer system instead of having a directory server.

## References

- [1] *Japanese Police target users of Tor anonymous network*  
<http://www.bbc.com/news/technology-22248692>
- [2] M. Wright, M. Adler, BN. Levine, C. Shields *An analysis of the degradation of anonymous protocols*, 2002.
- [3] Shamir, A. *How to share a secret*. Common.ACM 22 (11) (1979) 612-613.
- [4] Overlier L., Syverson P. *Locating hidden servers*. In Security and Privacy, 2006 IEEE Symposium on (pp. 15-pp).
- [5] ssd, *The Surveillance Self-Defense* <https://ssd.eff.org/tech/tor>
- [6] Cryptopp, <http://www.cryptopp.com/benchmarks.html>.
- [7] Dingledine R., Mathewson N., Syverson P. *Tor: The second-generation onion router*. Naval Research Lab Washington DC, (2004).
- [8] Syverson P. *Practical Vulnerabilities of the Tor Anonymity Network*. Advances in Cyber Security: Technology, Operation, and Experiences, (2013).
- [9] Kane A. M. *Another Tor is possible* Cryptology ePrint Archive (2014), 787.
- [10] 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).
- [11] Parmar K., Jinwala D. *A Novel Approach for Verifiable Secret Sharing by using a One Way Hash Function*. arXiv preprint arXiv:1203.3620.
- [12] Panchenko A., Niessen L., Zinnen A., and Engel T. *Website fingerprinting in onion routing based anonymization networks*. WPES, page 103-114. ACM, 2011.
- [13] Wikipedia, [http://en.wikipedia.org/wiki/Shamir's\\_Secret\\_Sharing](http://en.wikipedia.org/wiki/Shamir's_Secret_Sharing)

## 5 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  $\parallel$  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[13]. 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 a file  $file_1$  where she writes  $HM, HD_0, HD_1, HD_2, HD_3, HD_4, HD_5$  and the time of request. She authenticates itself and sends to the directory server  $file_1, a_1$  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  $file_1$  which becomes  $file_{1signed}$  and it takes at random 6 members of Tor network (Cosette, Maruis, Fantine, Hugo, javert, Valjean).

The directory sends a secret share to each of them (Cosette, Maruis, Fanny, Hugo, javert, Valjean). For example in the message of Cosette we can see:

$$E(Pub_{KeyCosette} : (3, 2578) \parallel HM).$$

The directory server sends  $file_{1signed}$  to Alice and deletes the secret shares and logs concerning Alice.

### Step 1:

Alice encrypts with the public key of Zola, the following elements:  $file_{1signed}$  and the message  $m$  for Bob.

$$C_1 = E(Pub_{KeyZola} : file_{1signed} || m).$$

After that, Alice encrypts with the Macquart's public key the file  $C_1$  which is the message for Zola.

$$C_2 = E(Pub_{KeyMacquart} : C_1).$$

Alice then encrypts with the public key of Rougon, the file  $C_2$  which is the package for Macquart.

$$C_3 = E(Pub_{KeyRougon} : C_2).$$

Alice sends  $C_3$  to the first relay.

**Step 2:**

Rougon will decrypt with his private key, to find the encrypted message for Macquart, and Rougon will send to Macquart  $C_2$ .

**Step 3:**

Macquart will open the message with his private key, to discover the encrypted message for Zola. Macquart will send to Zola  $C_1$ .

**Step 4:**

Zola will open the message with his private key, in order to find  $file_{1signed}$ , and the encrypted message for Bob.

After that he will check in  $file_{1signed}$  if  $HM$  matches with the hash of the message that Alice is trying to send to Bob, he will also verify that the signature and the timestamp are correct.

If all verifications are conclusive, he will send  $m$  to Bob and he will keep in a secure storage  $file_{1signed}$ . Otherwise, if  $SHA1(m)$  is different from  $HM$ , or if there is something wrong with the signature of the DS or the timestamp, 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  $file_{1signed}$  which he has recorded, the file where  $HM$  is equal to  $SHA1(m_1)$ .
3. Zola will broadcast to all Tor users, the anonymity revocation request and hashes present in  $file_{1signed}$  (i. e.  $HM, HD_0, HD_1, HD_2, HD_3, HD_4, HD_5$ ).
4. Tor users will study the request of that organization, those convinced by this demand will send their shares.
5. We assume that Valjean, Maruis and Fantine are not convinced by the arguments of the organization, Cosette, Javert and Hugo are convinced and will send (5, 4414), (6, 5614) and (3, 2578).
6. The organization must ensure that the hash of the shares received are present in  $file_{1signed}$  before reconstituting Alice's IP address using the Lagrange polynomial.