World Journal of Engineering Research and Technology
WJERT
www.wjert.org
SJIF Impact Factor: 4.326

A STUDY ON PRIVACY PRESERVING AUTHENTICATION SCHEME USING PAIRING BASED ONION ROUTING

Afreen Fatima Mohammed*[1] and Dr. A. Kanaka Durga[2]

[1]Assistant Professor, Department of Information Technology Institution: SCETW, Hyderabad Institution: SCETW, Hyderabad.

[2]Professor, Head of IT Department, Department of Information Technology Institution: SCETW, Hyderabad Institution: SCETW, Hyderabad.

Article Received on 08/08/2017 **Article Revised on 28/08/2017** **Article Accepted on 18/09/2017**

***Corresponding Author**
Afreen Fatima Mohammed
Assistant Professor, Department of Information Technology Institution: SCETW, Hyderabad Institution: SCETW, Hyderabad.

ABSTRACT

One of the major concern while sending data over the network is to preserve the identity of the user and protect the entire data traffic flowing from the user. The data send by the user will go through numerous intermediate nodes or routers. Even if one of these nodes get affected or compromised by the attacker, the entire data flowing through it will be identified by the attacker as well as the user's identities will be known to them. This paper discusses about pairing-based Onion Routing that supports non-interactive key arrangement scheme. It supports forward secrecy and provides relatively less computation and communication than the telescoping approach used by the Tor. These properties ensure that pairing-based onion routing protocol enables the anonymity networks to scale gradually and hence preserving the privacy of a data send by the user.

KEYWORDS: Tor, Telescoping, Pseudonyms, Forward Secrecy, Private key generator, Private key validity period, Master key Validity period.

1. INTRODUCTION

Onion Routing provides anonymous connections over the network. The message is encrypted in the form of layers, and is routed through numerous onion routers or OR's, often known as

intermediate nodes and the message is referred as Onion. These nodes relay incoming traffic from users. Each node will remove one layer from an onion through decryption. Hence the data to be transmitted is protected in these layers of onion and data privacy is achieved.

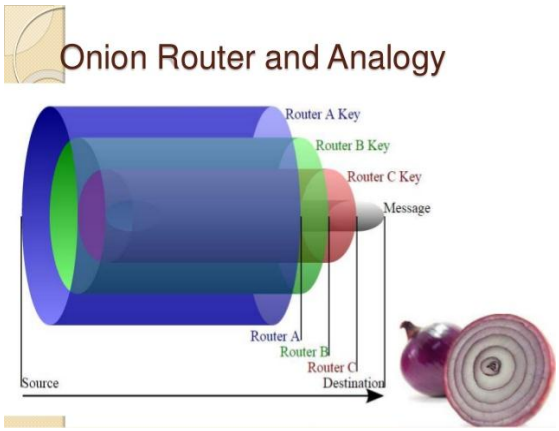

Fig 1.1: Message encrypted in the form of Onion.

The above figure shows that how message is encrypted in the form of Onion is taken from.[21]

The network of onion routers consist of a path, randomly selected by the user. This path forms a circuit which is a sequence of nodes that will route the incoming traffic. After the circuit is constructed, each node in the circuit shares a symmetric key with the user, which will be used to encrypt the layers of onion.[2] The drawback of this approach is that it does not support forward secrecy, which means that if a user constructs a circuit containing a sequence of nodes A,B and C, and if A is malicious, then A will record the entire traffic flowing through it, and later it may compromise B and then compromises C. This will lead to the revelation of an entire route to A, since it is malicious. The node A will learn who the user has communicated with. Hence the privacy among the users are altered. The solution to this problem is to frequently contact the directory server, to access authentic keys. This process is often called Telescoping, as discussed in,[2] where the Circuits are constructed incrementally.The idea is to use the node's public key only to initiate a communication during which a temporary session key is established via the Diffie-Hellman key exchange. One of the known Onion routing system is Tor, which has almost 1000 onion routers and hundreds of thousands of users.[1] Tor constructs circuits in this way, using the Tor authentication protocol (TAP).[2]

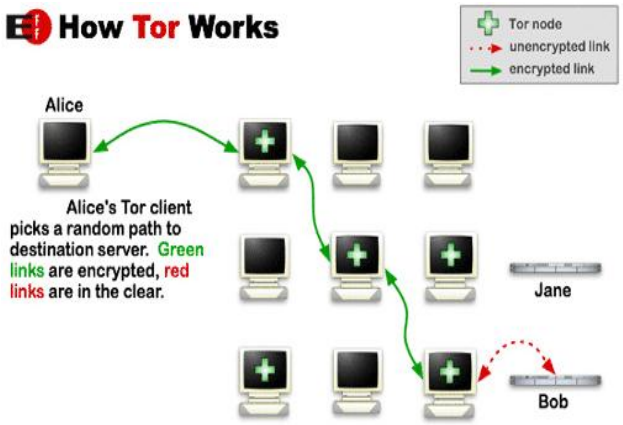

Fig 1.2: Working of Tor.

The above figure has been taken from[22] shows the working of Tor in a network when Alice and Bob wants to be anonymous.

This paper proposes a new scheme for preserving the privacy using Onion Routing that supports Forward Secrecy by building circuits in a single pass. It uses Pairing based Onion Routing scheme which make uses of OR's identities without repeatedly contacting the central directory server, hence providing forward secrecy without telescoping.

2. RELATED WORK

Over the years, large number of anonymity networks have been proposed and some have been implemented. Common to many of them is onion routing, a technique whereby a message is wrapped in multiple layers of encryption, forming an onion. As the message is delivered via a number of intermediate onion routers (ORs), or nodes, each node decrypts one of the layers, and forwards the message to the next node. This idea goes back to Chaum[3] and has been used to build both low-latency and high-latency communication networks. Discussions related to Formalizations and security of onion routing has been be found in.[4,5,6,7]

Some of the examples of Onion Routing systems are discussed in.[8,9,10,11,12] An onion routing system is usually constructed by arrangimg a collection of nodes that will relay traffic for users of the system. Tor is one of the largest onion routing system which has approximately 1000 onion routers and hundreds of thousands of users.[1] These numbers and their growth demand for anonymity online.

To use a network of onion routers, users randomly choose a path through the network and construct a circuit-a sequence of nodes which will route traffic. After the circuit is constructed, each of the nodes in the circuit shares a symmetric key with the user, which will be used to encrypt the layers of future onions.

In the original Onion Routing project[13,14,7] (which was superseded by Tor)circuit construction was done as follows. The user creates an onion where each layer contained the symmetric key for one node and the location of the next node, all encrypted with the original node's public key. Each node decrypts a layer, keeps the symmetric key and forwards the rest of the onion along to the next node and forward secrecy is not supported (as defined in[15]). As discussed in,[2] when a circuit consisting of nodes A, B, C, is formed and on eof the node, A is malicious, then A records the entire traffic, and at a later time compromises B (at which point he learns the next hop is C), then compromises C, the complete route is known, and A learns who the user has communicated with. A possible fix for this problem is to frequently change the public keys of each node. This limits the amount of time A has to compromise B and C, but requires that the users of the system frequently contact the directory server to retrieve authentic keys. Later systems constructed circuits incrementally and interactively. This process is sometimes called telescoping. The idea is to use the node's public key only to initiate a communication during which a temporary session key is established via the Diffie-Hellman key exchange. Tor constructs circuits in this way, using the Tor authentication protocol (TAP) discussed in.[16]

Forward secrecy is the main advantage of telescoping, but telescoping also handles nodes that are not accepting connections; if the third node is down during the construction of a circuit, for example, the first two remain, and the user only needs to choose an alternate third. Information about the status and availability nodes is therefore less important.[2] The drawback of telescoping is the cost of establishing a circuit of length ℓ requires $O(\ell 2)$ network communications, and $O(\ell 2)$symmetric encryptions/decryptions.[2]

The efficiency of telescoping-based circuit construction is improved using a half-certified Diffie-Hellman key exchange,[18] as discussed by Overlier and Syverson.[17] They also define an efficient single-pass circuit construction and a few variants. The proposed variants offer different levels of forward secrecy, which is traded off against computation and communication. For example, their eventual forward secret variants use frequent rotation of nodes' public keys, presenting the same issues as the first generation onion routing; their

immediate forward secrecy variant uses the same amount of communication as the current Tor (O(ℓ2)), but less computation.

Privacy-preserving authentication schemes can be one- or two-way, also referred to as unilateral or bilateral. After one-way authentication between Anonymous and Bob, Anonymous has confirmed Bob's identity and Bob learns nothing about Anonymous, except perhaps that he or she is a valid user of a particular system. In a two-way scheme, both users can confirm they are both valid users without learning who the other is.[2]

The work of Okamoto and Okamoto[19] presents schemes for anonymous authentication and key agreement. In Rahman et. al.,[20] an anonymous authentcation protocol is presented as part of an anonymous communication system for mobile ad-hoc networks. The protocols in both papers are complex, and limited motivation is given for design choices.[2] But the security of their proposed protocols in these two papers were not discussed.

3. ANONYMOUS-KEY AGREEMENT SCHEME

The following anonymous agreement sheme is used in this Pairing-based Onion Routing Protocol. As discussed in,[2] the users who want to preserve their privacy for communication has to register with Private Key Generator(PKG). PKG is treated as a trusted-authority, which generates private keys for clients using their identities and a master secret s. A participant can confirm that other participant is a client of same PKG, but cannot determine his identity. This is done in following steps:

STEP 1 Each user can generate many possible pseudonyms and the corresponding private keys. Suppose Alice, with (identity, private key) pair (ID_A, d_A), is seeking anonymity. She generates a random number r_A and creates the pseudonym and the corresponding private key:
$(P_A = r_A Q_A = r_A H(ID_A), r_A d_A = sP_A)$

STEP 2.In a key agreement protocol, she then sends the pseudonym P_A instead of her own identity to another participating client, who may or may not be anonymous.

STEP 3 For two participants (say Alice and Bob) with pseudonyms P_A and P_B, the shared session key is given as:
$K_{AB} = e(P_A, P_B)^{s} = e(Q_A, Q_B)^{rA\, rB\, s}$
where r_A and r_B are random numbers generated respectively by Alice and Bob.

STEP 4 If Bob does not wish to be anonymous, he can just use $r_B = 1$ instead of a random value, resulting in $P_B = Q_B$.

Two participants can exchange pseudonyms by performing a session key agreement. As discussed in,[2] two participants can also perform an Authenticated key agreement by modifying any secure symmetric-key based mutual authentication protocol and simply replacing their identities by their pseudonyms.

4. PAIRING-BASED ONION ROUTING PROTOCOL

Low-latency onion routing requires one-way anonymous key agreement and forward secrecy. In this section, we describe a new pairing-based onion routing protocol using the non-interactive key agreement scheme.

Pairing based Onion routing protocol has an advantage over the original onion routing protocol discussed in[14] as well as the protocol used in Tor.[15] It provides forward secrecy by building circuits in a single pass by telescoping method. The problem with the original onion routing protocol is that it involves regularly contacting OR's to access authenticated copies of ORs' public keys. Forward secrecy in such protocol is achieved by rotating the keys periodically as discussed in.[2] The problem with this approach is that if the public keys are changed all users must contact a directory server to retrieve the new authenticated keys. However, Pairing based onion routing protocol uses ORs' identities, which users can obtain or derive without repeatedly contacting a central server, thus providing practical forward secrecy without telescoping.

An onion routing protocol consist of service provider, set of onion routers, and users In this protocol, user builds a circuit in a single-pass but not incrementally via telescoping. The user selects ℓ ORs from the available pool and generates separate pseudonyms for communicating with each of them. The user computes the corresponding session keys and uses them to construct a message with ℓ nested layers of encryption. This process uses the Anonymous-key agreement scheme discussed above, ℓ times. The service provider works as the Private Key Generator for the ORs and provides private keys for their identities.

The Forward Secrecy supported by this protocol is discussed below:

4.1 FORWARD SECRECY

Forward Secrecy is achieved by means of two parameters:

Private key validity period (PKVP) and Master key validity period (MKVP).

4.1.1 PRIVATE KEY VALIDITY PERIOD

It specifies how much exposure time a circuit has, against compromises of the ORs that use it. Until PKVP elapses, OR's have enough information to collectively decrypt circuit construction onions, sent during this period. As PKVP elapses, ORs discard their current private keys and obtain new keys from the PKG's. This period can be short, just as the order of an hour.

4.1.2 MASTER KEY VALIDITY PERIOD

This period specifies the circuit's exposure time against compromises of the PKG which reveal the master secret s. Because changing *s* involves the participation of all of the ORs as well as the PKGs, the MKVP is longer than the PKVP, just like an order of a day.

As discussed in,[2] in the order of *t* of *m* distributed PKG, if atleast $m - t + 1$ PKG members are not compromised, no one will ever learn the value of a master secret.

5. PROTOCOL DESCRIPTION

As discussed in,[2] the usage of distributed PKG is proposed. Suppose PKG IS a single entity. The affect of PKG on the setup and key generation step are discussed below:

5.2.1 SET UP STAGE

During this stage, for a given security requirements, the PKG generates a digital signature key pair. It also generates a prime n, two groups G and G_T of order n and a bilinear map $e: G \times G \rightarrow G_T$.

PKG also chooses a full-domain cryptographic hash function:
$H: \{0, 1\}^* \rightarrow G^*$.

The PKG publishes all of these values except its private signature key.

5.2.2 KEY GENERATION STAGE

For Key Generation, the PKG generates a random master key for each MKVP $s \in Z_n^*$ and a random $U \in G$, and calculates sU.

The PKG publishes a signed copy of (v_m, U, sU),

where v_m is a timestamp for the MKVP in question. *U* is a common value to be shared by all users of the system.

For every valid OR with identity ID_i, and for every PKVP *v*, that overlaps with the MKVP, the PKG generates the private key $dvi = sH(v||IDi)$.

The PKG distributes these private keys, as well as a copy of the signed (*vm,U, sU*), to the appropriate ORs over a secure authenticated forward-secret channel. If an OR becomes compromised, the PKG can revoke it by simply no longer calculating its values of *dvi*.

PKG pre-computes the master keys and private keys in advance, and deliver them to the ORs in batches of any size from one PKVP at a time on up.

5.2.3 USER SETUP

During User setup**,** once every MKVP *vm*, each user must obtain a new signed tuple (*vm,U, sU*) from any OR or from a public website. Once every PKVP *v*, the user computes the following pairing for each OR *i* and stores the results locally:

$\gamma_{vi} = e(sU,Qvi) = e(U,Qvi)s$ *where* $Qvi = H(v||IDi)$

5.2.4 CIRCUIT CONSTRUCTION

As discussed in,[2] during a PKVP *v*, a user U chooses a set of ORs *(say A,B,...,N)* and constructs a circuit containing *U,A,B,....., N* with the following steps:

STEP 1 For each OR *i* in the circuit, the user generates a random integer $r_i \in Z^*_n$ and computes the pseudonym

$PUi = riU$ and the value $\gamma_{vi}^{ri} = e(U,Qvi)^{sr1.}$

From γ_{vi}^{ri} two session keys are derived: a Forward session key *KUi* and a Backward session key *KiU*. Finally, the onion is built and sent to A, the first OR in the circuit.

The pseudonym is given by:

$r_AU,\{B,r_BU, \{... \{N, r_NU, \{\emptyset\}KUN\}... \}KUB\}KUA$

Here $\{... \}K_{Ui}$ is symmetric-key encryption and ∅ is an empty message, which informs N that it is the exit node.

STEP 2 After receiving the onion, the OR with identity ID_i uses the received *riU* and its currently valid private key d_{vi} to compute $e(riU, dvi) = e(U,Qi)^{ris} = \gamma_{vi}^{ri}$

It derives the forward session key K_{Ui} and the backward session key K_{iU}. It decrypts the outermost onion layer $\{...\}K_{Ui}$ to obtain the user's next pseudonym, the nested cipher text, and the identity of the next node in the circuit. The OR then forwards the pseudonym and cipher text to the next node. To avoid replay attacks, it also stores pseudonyms. The process ends when an OR, the node N i.e the exit node gets ∅.

STEP 3 The exit node N then sends a confirmation message encrypted with the back-ward session key $\{Confirm\}K_{NU}$ to the previous OR in the circuit. Each OR encrypts the confirmation with its backward session key and sends it to the previous node, until the cipher text reaches the user. The user decrypts the cipher text layers to verify the confirmation.

STEP 4 If the user does not receive the confirmation in a specified time, she selects a different set of ORs and repeats the protocol.

User OR_A $^{OR}{}_{B\,ORC}$

$<U,Su>$ $<A,S_{QVA}>$ $<B,S_{QVB}>$ $<C,S_{QVC}>$

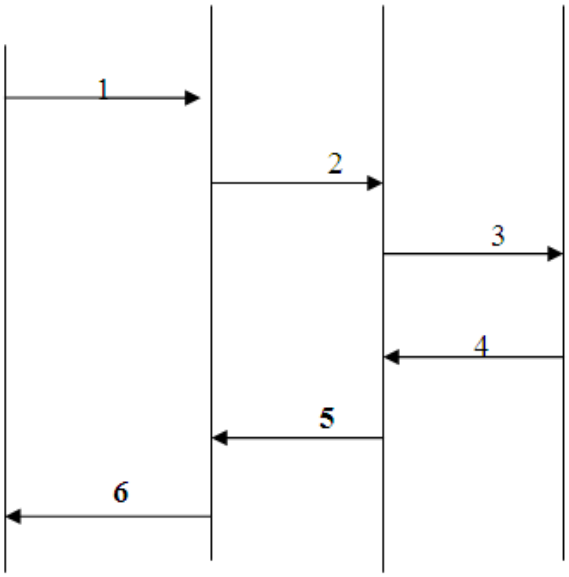

Fig 2: Circuit with three ORs.

In the above figure it is shown that how the message send by the user is encrypted in the form of layers and are send in the form of pseudonyms among ORSs. The following shows the pseudonyms which is send stepwise across three ORs**.**

1. $r_AU, \{B, r_BU, \{C, r_CU, \{\emptyset\}K_{UC}\}K_{UB}\}K_{UA}$

2. $r_BU, \{C, r_CU, \{\emptyset\}K_{UC}\}K_{UB}$

3. $r_CU, \{\emptyset\}K_{UC}$

4. $\{Confirm\}K_{CU}$

5. $\{\{Confirm\}K_{CU}\}K_{BU}$

6. $\{\{\{Confirm\}K_{CU}\}K_{BU}\}K_{AU}$

5.2.3 ANONYMOUS COMMUNICATION

After the circuit is constructed, communication proceeds in the same manner as Tor. A user builds a circuit with three ORs. The user sends onions through the circuit with each layer encrypted with the forward keys K_{Ui} and each hop decrypts one layer. Replies are encrypted at each hop with the backward key K_{iU}, and the user decrypts the received onion. Note that as an optimization, one or more messages can be bundled inside the original circuit construction onion, in place of ∅.

6. ADVANTAGES OVER FIRST-GENERATION ONION ROUTING AND TOR

The advantages of Pairing based Onion Routing over First Generation Onion Routing and TOR is discussed in the following section:

6.1 ADVANTAGES OVER FIRST-GENERATION ONION ROUTING

It is possible to achieve forward secrecy in first-generation onion routing by periodically replacing the public-private key pairs of the ORs.

Following the change, after getting authentic copies from the ORs, the service provider publishes signed copies of the new OR public keys. But it requires all users to regularly obtain fresh authenticated public key information for all ORs. In contrast, with Pair based Onion Routing Protocol, each user only needs to obtain the single authenticated value (vm,U, sU), and only once every MKVP. The user can then calculate the required γvi values on her own until the end of that period, thus reducing the load on the service provider. This load is further reduced by having the service provider never communicate directly with users at all, but only with the OR's.

As a consequence, Pairing based onion routing is a more practical solution for low-latency anonymous communication.

6.2 ADVANTAGES OVER TELESCOPING IN TOR

The Tor network uses telescoping approach based on the Diffie-Hellman key exchange to form an anonymity circuit. Pairing based Onion protocol requires an occasional private key generation for ORs to achieve forward secrecy. I t saves communication cost at every circuit construction by avoiding telescoping. The absence of telescoping provides flexibility to the user to modify a circuit on the fly. For example, suppose a user *U* has constructed a circuit *(U,A,B,... K,...,N)*. The user can bundle instructions to immediately replace K with K′ in the next message, while keeping the remaining circuit intact. The circuit would then be *(U,A,B,...., K',....., N)*.

7. ISSUES WITH THE PROPOSED SCHEME

The certifying authorities in the Tor system are less trusted than the PKG in our scheme. With a short PKVP and MKVP (compared to the keyr eplacement period in Tor), our PKGs (any *t* of them) need to be online with greater reliability. Further, if fewer than *t* are available, the whole system is compromised after the current batch. It is also possible for *t* malicious PKGs to passively listen to all of the traffic as they can compute private keys for all ORs. A geographically distributed implementation of *m* PKGs certainly reduces this possibility. To passively decrypt an OR's messages, an adversary of the Tor system must know the OR's private key, as well as the current Diffie-Hellman key (established for each circuit). In our proposed scheme, as it is non-interactive, an adversary who knows only the OR's private key can decrypt all of the messages for that OR. This may be an acceptable trade-off, considering the advantages gained from the non-interactive protocol.

7.1 SYSTEMS ISSUES

In this section, we describe how components of an onion routing system such as Tor would behave in a pairing-based setting. To implement pairings, we must choose groups where pairings are known, and are efficiently computable. Once these groups are fixed we can estimate the computational cost required to construct a circuit. The next section will compare the cost of our scheme to the cost of setting up a circuit with Pairing-based Onion Routing.

7.2 COST OF BUILDING A CIRCUIT WITH PARING-BASED ONION ROUTING

In order to create a circuit of length ℓ with our scheme, the user must choose ℓ random elements r_i of Z_n^*. User should not reuse these values. She then computes $r_S U$ and $\gamma S^{rs,}$ and derives the forward and backward keys K_{US} and K_{SU} from $\gamma S^{rS,}$ for each server S in the

circuit. Each server computes $e(r_S U, dS) = \gamma S^{rS}$ for its current private key d_S and derives K_{US} and $K_{SU.}$

User creates one message and sends it to the first server in the chain. This server decrypts a layer and sends the result to the second server in the chain, and so on, for a total of ℓ hop-by-hop encrypted messages. At the end of the chain, the last server replies with a confirmation message that travels back through the chain, producing ℓ more messages, for a total of 2ℓ.

8. CONCLUSIONS

Pairing-based Routing protocol creates circuits in a Single-pass, and also provides forward secrecy, thereby preserving the privacy. It requires less computation and communication than the corresponding protocol in Tor, and reduces the load on the network support infrastructure. Thus this enhances the scalability of low-latency anonymity networks and such anonymity networks can be grown gradually.

REFERENCES

1. The Tor Project. Tor: anonymity online. http://tor.eff.org/. Accessed M, February 2007.
2. “Pair based Onion Rouitng “, David R. Cheriton School of Computer Science, University of Waterloo, Aniket Kate, Greg Zaverucha, and Ian Goldberg.
3. D. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM, February 1981; 4(2): 84–88.
4. J. Camenisch and A. Lysyanskaya. A Formal Treatment of Onion Routing. In Advances in Cryptology—CRYPTO 2005, Lecture Notes in Computer Science 3621, Springer-Verlag, August 2005; 169–187.
5. S. Mauw, J. Verschuren, and E. de Vink. A Formalization of Anonymity and Onion Routing. In ESORICS 2004, Lecture Notes in Computer Science 3193, Springer-Verlag, September 2004; 109–124.
6. B. Moller. Provably Secure Public-Key Encryption for Length-Preserving Chaumian Mixes. In CT-RSA 2003, Lecture Notes in Computer Science 2612. Springer-Verlag, April 2003.
7. P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an Analysis of Onion Routing Security. In Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability, Lecture Notes in Computer Science 2009, Springer-Verlag, July 2000; 96–114.
8. W. Dai. PipeNet 1.1. Post to Cypherpunks mailing list, November 1998.

9. R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In Proceedings of the 13th USENIX Security Symposium, August 2004.
10. M. J. Freedman and R. Morris. Tarzan: A Peer-to-Peer Anonymizing Network Layer. In Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS 2002), Washington, DC, November 2002.
11. M. Reed, P. Syverson, and D. Goldschlag. Anonymous Connections and Onion Routing. IEEE Journal on Selected Areas in Communications, May 1998; 16(4): 482–494.
12. M. Rennhard and B. Plattner. Introducing MorphMix: Peer-to-Peer based Anonymous Internet Usage with Collusion Detection. In Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2002), Washington, DC, USA, November 2002.
13. 27. M. Reed, P. Syverson, and D. Goldschlag. Anonymous Connections and Onion Routing. IEEE Journal on Selected Areas in Communications, May 1998; 16(4): 482–494.
14. D. Goldschlag, M. Reed, and P. Syverson. Hiding Routing Information. In Proceedings of Information Hiding: First International Workshop, Lecture Notes in Computer Science 1174, Springer-Verlag, May 1996; 137–150.
15. R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In Proceedings of the 13th USENIX Security Symposium, August 2004.
16. I. Goldberg. On the Security of the Tor Authentication Protocol. In Proceedings of the Sixth Workshop on Privacy Enhancing Technologies (PET 2006), Lecture Notes in Computer Science 4258, Springer-Verlag, June 2006; 316–331.
17. L. Øverlier and P. Syverson. Improving efficiency and simplicity of Tor circuit establishment and hidden services. In Proceedings of the 7th Privacy Enhancing Technologies Symposium (these proceedings), 2007.
18. A. Menezes, P. Van Oorschot, and S. Vanstone. Handbook of Applied Cryptography. CRC Press, 1st edition, 1997.
19. E. Okamoto and T. Okamoto. Cryptosystems Based on Elliptic Curve Pairing. In Modeling Decisions for Artificial Intelligence—MDAI 2005, Lecture Notes in Computer Science 3558, Springer-Verlag, July 2005; 13–23.
20. S. Rahman, A. Inomata, T. Okamoto, M. Mambo, and E. Okamoto. Anonymous Secure Communication in Wireless Mobile Ad-hoc Networks. In First International Conference on Ubiquitous Convergence Technology (ICUCT2006), December 2006.
21. https://en.wikipedia.org/wiki/Onion_routing.
22. https://www.google.co.in/OnionRouitng/images/how TOR works.