ISSN: 2321-7782 (Online)
Volume 2, Issue 5, May 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Privacy for Interactive Web Browsing: A Study on Anonymous Communication Protocols

S.Shakila[1]
Research Scholar
School of Computer Science and Engineering
Bharathidasan University
Tiruchirappalli – India

Dr. Gopinath Ganapathy[2]
Professor and Head
School of Computer Science and Engineering
Bharathidasan University
Tiruchirappalli – India

Abstract: ***Anonymous communications plays an indomitable role in today's technological scenario especially for a shared public network environment. It aims not only to preserve security but also preserves the privacy and integrity of the content. Several anonymous systems have been designed, implemented and used by journalists, military and other government organizations to protect their identity. Research work on Anonymous systems was initiated by David Chaum's MIX network and DC-net. In this paper low latency anonymous protocols were considered for analysis. Onion routing is one of the low latency anonymous communication protocols which is resistant to traffic analysis and provides unlinkability between the responder and the messages passed. Tor is the circuit based second generation onion router which relies on distributed overlay network and on onion routing.***

Keywords: Unlinkability, Unobservability, Traffic analysis, Pseudonymity, Finger printing, Low latency, Mix networks.

I. INTRODUCTION

With the tremendous growth of Internet users and rapid advancement in the technology over the years paved way for the people to depend on networked distributed systems to carry out their daily activities. However, people show great concern over their personal details being tracked by the third parties. It can be forthrightly said that, privacy is a major issue on the web because the IP address of the user, domain name , organization, referred website, information requested etc. are being advertised by the browser. This information is available to end servers, local system administrator, and to other third parties posing a serious threat to the privacy. Cookies are another violation of privacy. Though end-to-end encryption protects the data from intruders, anonymity of communication is not preserved. Privacy preservation of source and destination has become a prerequisite for government organizations to serve various applications like electronic voting, auctions, incident reporting, witness protection etc.

In large organizations sensitive information are handled and so they want all their communications to remain anonymous rather than being intruded by the adversaries which is detrimental to their development. Therefore, all privacy preserving interactions must be capable of communicating anonymously. To achieve anonymous communication, [1] introduced Mix-networks that used layered encryption approach by wrapping the messages in several layers and routing through intermediate nodes. Each intermediate node peels a layer of encryption and forwards the message to the next node in random order. This process is repeated until all layers are removed. To achieve anonymity on the Internet a number of techniques have been proposed [5,11,14,15,16]. These techniques operate at the application layer and rely on overlay network that uses cryptography to route messages abstracting the content or destination of message from adversaries. Anonymous communication systems are categorized into cryptosystem-based schemes, routing-based schemes, broadcasting-based systems and peer-to-peer communication systems.

Regardless of various attempts that have been made to achieve anonymity, onion routing is used for establishing anonymous communication over a public network. In onion routing, instead of establishing a direct connection between the two hosts that want to communicate, the connection will be routed through a set of routers called onion routers which allows messages to travel from source to destination through an unpredictable path, allowing communication to be anonymous. Every node will only have information about its previous hop and the next hop i.e., the person who he/she is communicating with and the person with whom he/she is supposed to communicate. The rest of this paper is organized as follows: In Section 2, we describe the terminology used in anonymous communications. The various anonymization techniques are described in Section 3. In Section 4 an overview of anonymous communication systems is given. A Comparative study on various anonymous communications is depicted in Section 5. Section 6 illustrates the research directions to optimize the performance of widely used Tor protocol. Section 7 concludes the paper.

II. TERMINOLOGY IN ANONYMOUS SYSTEMS

The preliminary concepts of anonymity are explained in this section which is used in the subsequent sections to measure the degree of anonymity.

A. Anonymity and Pseudonymity

Anonymity is one of the desirable property of communication that must allow a user to use a service or application without revealing his identity. Pfitzmann and Hansen [2] informally defined Anonymity as, the state of not being identifiable within a set of subjects, the anonymity set. Pseudonymity is the process where pseudonyms are used as a subject identifier. Its main goal would be to allow users to run services or access resources without having to reveal their own identity. Pseudonyms are generally dynamic identifiers, or names of the subjects that are hard to be linked to the real identities without the shared secret keys.

B. Unlinkability and Unobservability

The goal of an anonymous system is to provide unlinkability between received messages and their senders(sender anonymity), and sent messages and their recipients(receiver anonymity) so that when adversaries are observing the network, they can solely see the senders and recipients but cannot see who is communicating with whom[3]. To provide this property, a system must generally be allowed to delay messages arbitrarily, or inject cover traffic.

The anonymity of a participant is naturally preserved as long as he doesn't interact with others. Only when he communicates with others his identity is revealed. Hiding the identity of participant implies, preserving the anonymity of the participant. Achieving anonymity in open environments such as the Internet is a challenging issue. Encryption alone cannot preserve the anonymity of communication, since the identity of participants can be easily inferred from the data that is used to support the communication.

Undetectability or Unobservability is the property whereby a subject can use a resource without giving opportunity to an adversary to determine what it is being used.Alice and Bob take part in some communication, but no one can tell if they are transmitting or receiving messages.

Pfitzmann and Hansen[2] defined the Unlinkability between senders and recipients in an anonymity system as relationship anonymity. Relationship anonymity can be simply referred to as; an adversary observing the senders and recipients in the network is unable to discern who is communicating with whom.

III. VARIOUS ANONYMIZATION TECHNIQUES

Anonymization in network can be achieved using the following techniques

- **Aggregating**: Packets are aggregated more efficiently to hide the communication relationship.

- **Batching**: In this approach first, the Packets which arrive at the Mix are buffered and then sent out in a batch. The batch transmission can be triggered on timer-based, threshold-based, or, a combination of both.
- **Reordering**: The order of dispatching the packets will be randomly arranged so that less information about arrival order can be inferred from dispatch order and vice versa.
- **Pooling**: In a pooling based Mix system, an arrival packet will be sent out with a probability each time the packet has a chance to be sent out.
- **Padding:** Mix system can be padded with dummy traffic to break the correlation between incoming and outgoing packets.
- **Rerouting**: Anonymity networks prefer a longer path which is chosen randomly by the source router instead of the conventional shortest path.
- **Layering:** Anonymity networks can use a layered approach to abstract the identity of the intermediate notes along the route using public key cryptography.

IV. Overview of Anonymos Communication System

The anonymous communication systems can be divided in two main groups: high latency and low-latency anonymous communication systems.

A. High Latency Anonymous Systems

High latency anonymity systems can be used only for non-interactive application. Although they provide strong anonymity they generate delays of several hours or so which is applicable only for sending emails. High-latency anonymity systems are also referred to as message-based systems.

1) Penet Remailer

It is a single proxy that provided email services only. If a user sends a message to the service, an alias is generated for the sender and the real email address of the user and the alias are saved in a database. It is similar to creating dummy numbers for students in University Examinations. Any identifying information from the message is stripped of by the remailer and forwarded to the intended user. As it fails to provide enough security and anonymity it was shut down due to legal issues[4].

2) Babel Remailer

It was built with Perl Programming Language and PGP(Pretty Good Privacy) is used for encryption. It allows email users to converse anonymously. It is vulnerable to many attacks as it depends solely on PGP for cryptographic computations. It takes atleast 24 hours to send a mail that makes it inefficient for bulk transfers[5].

3) Chaum'sMIX

This is the building block of all modern anonymous system introduced by Chaum. Chaum [1] used relay nodes or proxies called *Mixes*, which collects packets from multiple users, groups and outputs in such a way that an external adversary cannot link outgoing packet to the sender. However using traffic analysis an observer may infer sender-receiver relationship. A Mix conceals the content of every packet using encryption techniques and then modifies the packet timing by delaying and reordering the packets. Thus the latency of transmitted packets gets increased due to the modifications on packet timing. On the other hand, if strict latency constraints are to be imposed on packets, the capabilities of Mix are controlled to reduce the anonymity of the outgoing packets.

In the original design by Chaum, for n inputs from n different users, the Mixing strategy has to wait until packets are received and send all the packets out in one batch. As the packets of a batch can be arbitrarily ordered, it is impossible for an

evesdropper to identify the source of any packet. This strategy has been subsequently improved upon to address delay constraints [6], and also extended to networks of Mixes [7].

Operation of a MIX:

Assume there is a Mix M with public-private keys and users A and B. To protect from traffic analysis M encrypts all communication by sending all messages in the same format and length. User A constructs a message for delivery to user B by appending a random value R to the message, seals it with user B's public key KB, appends B's address and then seals the result with the Mix's public key PKM.

So A sends: PKM (KB(message, R), B's address) to M. Upon receipt of message, M decrypts the payload with his private key, and retrieves the destination address B.

So M sends: KB(message, R) to B. Upon receipt, B decrypts the payload with his private key, and now he can read the message.

An adversary can trace the sender via two techniques: traffic analysis and man in the middle attack. He can easily compute the following: First, he analyses the packets being transmitted from A to M, then he can look at the packets being transmitted from M to B. Once this is examined, the eavesdropper will grab one of the messages and concatenate B's address. KB(message, R) adds B's address (KB(message, R), B's address) encrypts with M's public key PKM (KB(message, R), B's address) once he computes this, he can compare the packets that are arriving at M and trace A. To overcome this, a nonce or random number is added to the message that a sends to M and the nonce will be removed by the Mix as it reaches its destination.

The performance of a Mix strategy is quantified under strict delay constraints using entropy-based metric of anonymity. Packets are transmitted to a Mix according to independent Poisson process. Every received packet can be delayed by the Mix up to a maximum of T seconds. As the Mix uses encryption to perfectly de-correlate the contents of incoming and outgoing packets, the eavesdropper can only observe a single departure process. Using the packet timing in the arrival and departure processes, the eavesdropper's goal is to identify the source of each departing packet. The anonymity of a Mixing strategy is measured using the normalized entropy of the a posteriori probability distribution and the maximum anonymity is characterized as a function of arrival rate and delay constraint [8].

The performance of Mix network is characterized by delaying and reordering of messages and the use of dummy traffic. These performance components have to be appropriately balanced making it difficult for an adversary to correlate the incoming and outgoing messages. The bandwidth consumption increases when the data is delayed longer at the proxy affecting the end-to-end performance. When real messages are not available dummy messages are sent instead to make the messages indistinguishable by an adversary. Dummy messages consists of random bits which cannot be distinguished from encrypted real messages providing a high level of resistance against traffic analysis attacks. The relative amount of dummy messages is increased when there are few users and the download times becomes shorter. Fewer real messages results in an increase of dummy messages and vice versa[9].

If the size of sender anonymity set is less than the population of total users, the information gathered from traffic observation is sufficient to deduce all communication relationships between senders and receivers using the Mix. Repeated communication can also reduce anonymity. Chaum's Mix needs very large buffer if long data streams are transmitted and so it cannot perform well. If buffer size is limited then any batching strategy may reveal the source identity[10]. The actual traffic flow pattern may be masked by inserting dummy traffic, however the use of dummy packets can reduce the network throughput. On the other hand, if the buffer size is fixed, then the Mix has to wait for at least one packet to arrive before the buffer is full. If packets arrive before the buffer is full, then one packet from each user is selected, randomly ordered and sent in succession. To achieve maximum degree of anonymity dummy packets are relayed by subsequent nodes as if they are data packets causing a drop in the rate of data packets resulting in a tradeoff between anonymity and throughput[11].

Limitations of Mix

Chaum's *Mixes* requires public-key cryptography, which is computationally expensive and slow for synchronous communication. Mix is vulnerable to timing attacks. It requires lots of traffic and adds latency to network flows. Mix is vulnerable to collaboration by first and last *Mixes* in the chain (timing attacks).Though Mix based system is operational, it is incapable to deliver better performance due to bandwidth overhead. The operation of anonymity network should be efficient to provide a good balance between the level of anonymity it offers and cost in terms of bandwidth overhead. There is no sender anonymity from the first *Mix* or system administrator. Using dummy traffic to obfuscate real traffic and buffering makes it inefficient.

B. *Low latency anonymity systems*

Mix based systems use flushing algorithm[12] to decide which message to forward to the next destination and so it introduce large delays that is tolerable only for email. However real time and interactive applications like web browsing needs low latency that is vulnerable to traffic analysis attacks.

1) Crowd

To increase the privacy of web transactions a system called Crowds is implemented. The idea behind this approach is hiding one's action within the actions of many other users. According to this approach, a user wishing to execute web transactions first joins a crowd of users. A member from among the crowd is randomly selected and the user's request is passed. That random member may either submit the request directly to the end server or forward it to another member chosen at random. This is repeated until the request eventually reaches the destination. Thus the end server is prevented from knowing the identity of the initiator of the request. The identity of the initiator is not distinguished even by the members of the crowd. Crowds do not use public cryptography and was designed only for anonymous web browsing [13].

It is a collection of users who run a special process called *jondo* which acts as a proxy. Some of the *jondos* may be corrupted or controlled by an adversary. Each corrupted *jondo* may observe messages sent to it but they cannot observe the messages transmitted on the links between honest *jondo*s. An honest *jondo* has no way of determining a particular *jondo* as corrupted or honest and corrupted *jondo*s can also try to compromise the honest users' anonymity to share observations. All *jondo*s must register with a centralized special server called *blender* which plays the role of coordinating the activities of the crowd. The *blender* is responsible for authenticating new users into the crowd. After authentication it then sends other *jondo*s encompassing the crowd to the new *jondo*.

An interesting property of Crowds is that it may lead to negative and positive consequences when a member of a crowd also submits requests of other users. Either the user may be incorrectly suspected of originating the request or it may offer some degree of deniability. However Crowds fail to protect a user's anonymity if the content of her web transactions reveal her identity to the web server. This executable web content for eg., ActiveX controls, Java Applets may open network connections directly exposing the user to the end server bypassing the crowd. Therefore, Java Applets and ActiveX controls should be disabled in the browser before using Crowds [13].

The degree of anonymity offered by crowds depends on the path setup protocol [14]. The path setup protocol establishes a routing path through the crowd, so that all subsequent communications between member and the web server is routed along it. This protocol is executed each time when the crowd members wish to establish an anonymous connection to a Web server and this is called a session. Whenever the crowd membership changes, the existing path are scrapped and a new protocol session is executed to create a random path to the destination.

As pair wise encryption is used to establish communication between the *jondo*s, the incoming message is not linked to the outgoing message. This is due to the fact that different keys are used for encrypting the message for transmission to the next

hop. Receiver anonymity is not achieved as every *jondo* has to know the receiver address for transmission to the designated receiver. However, as the messages are encrypted hop-to-hop a local eavesdropper is not able to reveal neither the identity of the receiver nor the initiator. Thus sender-receiver unlinkability is guaranteed in the Crowds system.

In case of collaborating *jondo*s the sender anonymity of the Crowds system is probable innocence. If the network grows to infinity, absolute privacy is achieved because the probability for revealing the identity of the receiver gets lowered if the number of malicious nodes remains constant. The use of padding to keep the packet size fixed throughout the communication may reveal information to the adversary via the contained amount of padding. The constant padding can have an additional decreasing effect on the chosen threshold [14].[15] suggest bidirectional anonymous communication for Crowds by building a virtual channel and assigning distinct virtual channel numbers for each connection. The channel number for the first *jondo* is first chosen by the initiator and it is stored in the data packet. Thus the first *jondo* knows to which *jondo* it has to forward the message and which channel number to use. This offers the following advantage: if an attacker gets two different packets, he cannot tell if both belong to the same communication. If the receiver wants to send data backwards to the original sender (initiator), he only has to use the channel number which is common for him and his predecessor.

Crowd poses threats due to three types of attacks; the end server is the web server to which the web transaction is directed. A local eavesdropper on the local area network, such as an administrator may monitor web usage at a local firewall. However, if the same Local Area Network (LAN) also serves the end server, then the eavesdropper is effectively global, and no protection is provided against it. If a local eavesdropper and the end server to which the user's request is destined, collaborate in an attack, then neither sender anonymity nor receiver anonymity is achieved.

Crowds cannot defend against denial-of-service attacks by rogue crowd members. A crowd member could accept messages from other crowd members and refuse to pass them along. Such denial-of-service is detrimental if a member fails benignly or leaves the crowd. As a result, these attacks are detectable. Crowds are susceptible to active attacks where crowd members substitute wrong information in response to web requests that they receive from ludicrous crowd members. Such attacks are inherent in any system that uses intermediaries to forward unprotected information, but fortunately they cannot be utilized to compromise anonymity directly.

Crowds differ from Mixes in several ways. Sender anonymity is achieved against crowd members whereas Mix systems do not provide sender anonymity, but provide sender and receiver unlinkability [1]. Another difference is that Mixes provide sender and receiver unlinkability against a global eavesdropper whereas Crowds do not provide anonymity against global eavesdroppers. Another distinguishing feature of Crowds' is its efficiency when compared to Mixes. The length of a message routed through Mix grows proportionally to the number of Mixes through which it is routed that incurs a cost of n public key encryptions and n private key decryptions that makes it expensive. The advantage of crowds is that, the throughput of crowd increases as each user actively participates in the crowd in the sense that the load on each user's computer is expected to remain roughly constant as new users join the crowd.

2) Hordes

Hordes is a successor to Crowds developed by Levine and Shields[16]. The design aspect of Hordes is similar to Crowds but it depends on IP multicast to send messages to the initiator. They use the same features of crowds to route messages on the forward path but uses different scheme for reverse path routing .Group multicast address is used to specify the return address instead of initiator's IP address. The reason for using a group multicast address is that it makes it difficult to reveal the participants of the group. Another significant advantage of multicasting is to avoid Transmission Control Protocol (TCP) connection overhead gained during the return path that impedes the performance of the crowd. However, there are subtle differences between Crowds and Hordes. In Hordes messages are forwarded using a small random subset of other nodes and the size of subset chosen is that Hordes is not vulnerable to predecessor attack than crowd. Hordes use User Datagram Protocol

(UDP) for message passing contrast to Crowds which use TCP connections. Hordes need Congestion controlled transmission protocols. Hordes give better performance and the degree of anonymity offered by it is higher than crowd. The weakness of Hordes due to Internet multicast may be compromised using Internet Protocol Version 6(IPV6).

Since TCP connections cannot be established using multicast addresses, Hordes require a different transport mechanism – UDP(User Datagram Protocol). Forward connection uses TCP connection like crowd and encapsulates the multicast address and a random number to identify the return message within the data of the TCP payload. The destination server cannot interpret the encapsulated horde packet unless it is a horde and designated as a horde server. The responsibility of forwarding the server response vests with the responder, which take the payload and send it via the multicast address.

There are slight differences between Hordes and Crowds in the forward path and reverse path back to the client. Crowds use TCP for all communications whereas hordes use a combination of TCP and UDP. In a crowd the forward and reverse data paths follow the same set of *jondo*s, so a timing attack between a request and response may reveal information between the two. In Hordes, as the packets directly travel back to the client without going through the *jondo*s on the forward path, they are not subject to the timing attack. A horde offers minimal protection against timing attack since the response of an attack isn't sent to the forwarding *jondo*s, unless they are listening to the same multicast address. The success of timing attack decreases if more traffic flows through each *jondo*.

3) Onion Routing

Syverson introduced onion routing as a mean to establish an anonymously redirected encrypted path through a network, with full control of routing decisions and identity disclosure left in the hands of the sender. Onion Routing protects its communications against traffic analysis attacks. It examines data packets flowing over the network making it very hard for network observers (such as crackers, companies, and governments) to reliably learn who is talking to whom and for what purpose. It hides the source and destination of a packet, rather than the content of the packet. The content of the packet can also be encrypted using any form of cryptography prior to sending[17].

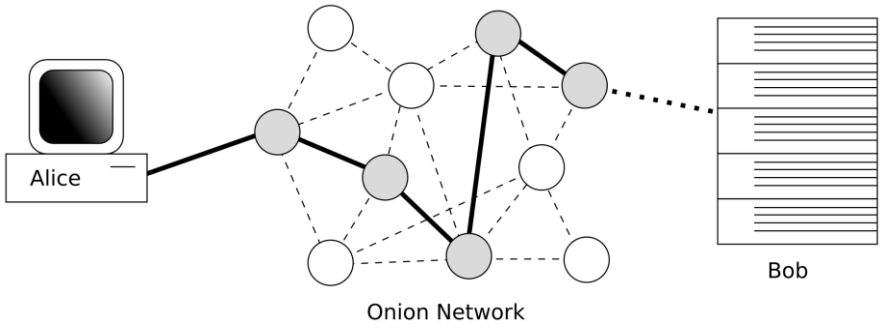

Fig. 1 Alice chooses a path through the onion network to communicate with Bob. Darkened nodes are the chosen proxy servers. Thick, solid lines are encrypted links; thick, dotted lines are unencrypted.

Onion Routing provides an anonymous socket connection through a proxy server. Since proxy is a well defined interface at the application layer, proxy servers can use many protocols in order to accommodate firewalls. In onion routing, rather than making socket connections directly to a responding machine, initiating applications establishes connections through a sequence of onion routers. The connection between the initiator and responder remain anonymous in Onion routing. Anonymous connections hide the identity of the source and destination from both outside eavesdroppers and compromised onion routers. If the initiator also wants to remain anonymous to the responder, then all identifying information must be removed from the data stream before being sent over the anonymous connection. Onion routers in the network are connected by permanent socket connections which are multiplexed. To set up an anonymous connection, the sequence of onion routers in a route is strictly defined. Each onion router can only identify the previous and next hop along a route. The messages passed along the

anonymous connection appear different at each onion router, hence data cannot be tracked even by the compromised onion routers.

Onion Routing's anonymous connections are protocol independent and exist in three phases: connection setup, data movement, and connection termination. The setup phase of Onion routing contains three logical layers: the application proxy (client proxy), the onion proxy (core proxy) and the onion routers. The application proxy gets the client request, strips the identifying information, some portion of request header, then builds and handles anonymous connections. The Onion proxy is the most important component because it knows the true source and destination of connections. It is involved in building the onion and managing it. Before building the onion, the onion proxy defines the route for anonymous connection based on the topology, link state of the network, the public certificates of nodes in the network and exit policies. The onion consists of different layers each of which can be decrypted only by a specific router who knows its public key. Each layer consists of information about the next hop, key material for backward and forward encryption, destination port and address, expiration time, key seed material etc. encrypted using the public key. The actual payload is encrypted using the symmetric key.

Once the onion is constructed, it is moved to the first onion router which uses its private key to decrypt the first layer of the onion. Each onion router along the route uses its private key to decrypt the layer to identify the next hop and sends the onion to the next router along the path. Since the size of the onion reduces as it nears the destination an attacker can infer details about the destination. To avoid this onions are padded at each onion router to maintain the size of the onion. Padding adds redundancy and is advantageous as it complicates traffic analysis, so that an attacker cannot infer location or other details of the destination by getting hold of an onion. The final onion router in the path connects to a responder proxy, which will forward data to the remote application.

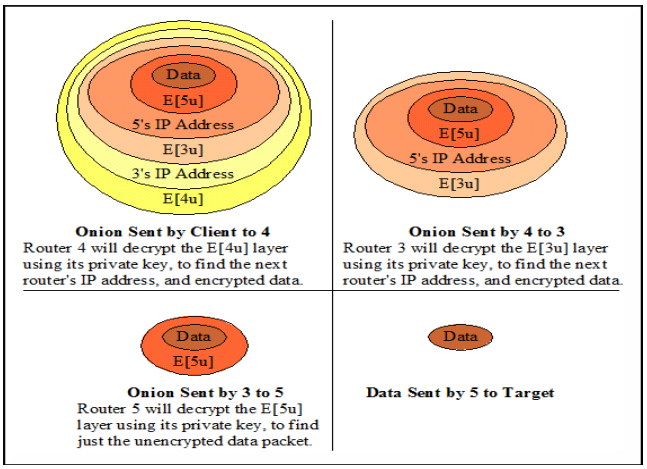

Fig. 2 showing the structure of the onion after every hop

Let the onion proxy select a random sequence of routers say 4,3 and 5 and constructs the onion as given in Fig. 2. It first encrypts the data packet with public key of 5 followed by public key of 3 and finally 4. So the encrypted data now looks like E4pu (3's IP address, E3pu ((5's IP address, (E5pu (recipient's IP address, data))))). The onion is sent by the client to onion router 4 which uses its private key to peel the outermost encrypted layer and finds the IP address of the next hop router 3. This is repeated until the onion reaches the last router 5.

To protect the onion from replay attack each layer of onion contains an expiration time so that an onion router is to ignore expired and replayed onions. Further if the connection breaks during the routing process then all the onion routers are informed via a destroy message. The final onion router is responsible for taking the response to the initiator for which it builds one layer of reply onion using its private key and sends it to the previous router. The previous router does the same task. The connection's previous router is identified using the connection identifier send earlier and sends reply onion again encrypted with its private key. Thus the reply onion finds its way back to the proxies of the initiator. The entire reply onion is decrypted by the onion

proxy with the public keys of the routers along the routing path. The response is then sent back to the client proxy and rendered appropriately to the application.

4) Tarzan

Tarzan [18] is a peer to peer overlay network based on the IP protocol to communicate anonymously. Each node has a set of neighbors to impersonate (mimics) the IP address of this node. Nodes with IP addresses from the same subnet are grouped together. To construct an anonymous circuit with an initiator, Tarzan chooses the first hop randomly from their set of mimics. Then the second hop will be selected from the set of mimics chosen by the first hop, and so on. In each hop, symmetric keys are generated and encrypted with public keys of the servers in the circuit, in a similar way to onion routing [17]. Like Crowds [13] all the users in the network relay traffic for other users. As the initiator of a circuit is also involved in exchanging mimic traffic, an adversary watching the node finds it difficult to identify it as the source.

Tarzan uses UDP for transport protocol as opposed to [13,17]. The main caveat of Tarzan is it operates at the Internet Protocol (IP) layer and requires applications using IP layer to replace with IP layer of Tarzan.

5) Tor

Tor [19] is the second-generation onion routing to enable anonymous communications on the internet. Tor is an anonymizing Internet proxy service that alternates the TCP traffic within chained, encrypted tunnels to circumvent traffic analysis. Users of the Tor network run an onion proxy on their machine to periodically negotiate a virtual circuit. Tor employs cryptography in a layered manner to ensure perfect forward secrecy between routers. Though people believe Tor for privacy, attackers may use this service to hide the true source and destination, or bypass a corporate security policy in order to view prohibited web sites. Tor is application independence and works at the TCP stream level. Tor is used in Internet Relay Chat (IRC), instant messaging and browsing the Web. To add privacy at the application layer when browsing the Web, Tor is coupled with Privoxy, a filtering proxy server.

Issues of Tor

Anonymity in Tor is not as strong as expected and cannot resist website fingerprinting attacks. Website fingerprinting [20] is an attack where a local attacker observes the encrypted data and tries to draw conclusions from certain features of the traffic, such as the volume of transferred data, the timing, or the packet sizes To hamper traffic analysis, padding is the most common technique used, where a certain amount of dummy data is appended along with original data before encryption. Padding is used to achieve fixed packet size which slightly reduces the detection rate compared to packets without padding. [21] Used camouflage as a counter measure to intentionally change the traffic pattern. Camouflage randomly loads a page (background page) along with the requested page to obfuscate traffic. When camouflage is used for Tor, the detection rate reduced from 55% to 3% and moreover it can be applied on the user side without modifying the anonymizing network.

To transfer massive amounts of data across the Tor network, the onion routers are run by volunteers using their own bandwidth at their own cost thus hogging the bandwidth of Tor network. Anonymous usage of SMTP (i.e., e-mail) can result in spam. Tor not only provides anonymity to client but also provides anonymity to servers. Tor is used to access hidden services hosted behind firewalls without requiring public IP address. The performance of Tor is influenced by the path selection algorithm and path length. An efficient path selection algorithm is required to optimize the trade-off between anonymity and performance. Tor uses random path selection algorithm using the bandwidth donated from users. Donated bandwidth may be limited by users causing bottlenecks for TCP throughput. As the path length increases, the number of routers is increased. Eventually the TCP throughput is decreased due to the high probability that a low-bandwidth Tor is chosen. There is a wide distribution of Tor routers leading to large round trip time and high packet drop rates. Large number of TCP connections shares

the Tor routers and links and hence contention for the shared bandwidth reduces the TCP connection's bandwidth lowering the overall throughput for a path through the Tor network [22].

Benefits of Tor

Tor provides perfect forward secrecy and uses guard nodes to improve source anonymity. Tor uses relay nodes to optimize the trade-off between performance and anonymity. Tor introduces hidden services which can be accessed only by the servers that are accessible by the Tor overlay. Despite the issues, it is the largest well deployed anonymity preserving service on the Internet publicly available to attract 500K to 900K users around the globe. There are more than 5000 Tor relays run by volunteers to optimize the performance.

V. COMPARATIVE ANALYSIS

Various high latency and low latency anonymous systems have been described in Section IV. The table below gives a characteristic analysis to ascertain the degree of anonymity and strength of security offered.

TABLE I

Comparative Analysis of the various Anonymous Systems

Type of Anonymous systems	Protection level	Privacy	Sender anonymity	Receiver anonymity	Unlinkability	Protocol used
Mix	Vulnerable to timing attack	No	No	Yes	Yes	TCP
Crowd	Vulnerable to DoS attack	No	Yes	No	Yes	TCP
Hordes	Minimal protection against timing attack	Yes for reverse path only	Yes	No	Yes	TCP – forward UDP- reverse
Tarzan	Minimal protection against traffic analysis	Yes	Yes	No	Yes	IP UDP-transport
Onion Routing	Vulnerable to Traffic analysis, DoS	Yes to first n-1 routers	Yes	Yes to first n-1 routers	Yes	No limits to types of protocols
Tor	Vulnerable to attack only if JavaScript, Applet, ActiveX are enabled	Yes to first n-1 routers	Yes	Yes to first n-1 routers	Yes	TCP

VI. RESEARCH DIRECTIONS

Anonymity is the property of being unidentifiable within a group. In the context of communication networks, like the Internet, there are several forms of anonymity, such as sender anonymity, receiver anonymity, and unlinkability. Several protocols have been developed that allow one to communicate anonymously over the internet. The Tor protocol shows high degree of anonymity than other protocols. Though a lot of research has been carried out to measure anonymity and to guard against potential attacks, there is no clear approach to model or establish them. Due to the wide use of Internet based applications, Hyper Text Transfer Protocol(HTTP) traffic comprises an overwhelming majority of the connections and it is unclear whether Tor can facilitate interactive web browsing. Tor designers emphasis on achieving low latency and reasonable throughput in order to allow interactive applications, such as web browsing, to take place within the network. The behavior of Tor may be modeled using Reinforcement Learning approach to analyze the degree of anonymity offered. The capabilities of the adversary can be studied using Support Vector Machines. The computation complexities and latency arising due to the layered cryptographic operations can be reduced using parallelizing techniques. Tor needs suitable blocking strategy to evade insecure protocols like Post Office Protocol (POP), Telnet and File Transfer Protocol (FTP) etc. as they may pave way for eavesdropping exit router to capture identifying information. A rule based blocking system can be used to block protocols that

commonly leak identifying information and should not be multiplexed over the same circuit with other non-identifying traffic. Path selection algorithm may be based on game theoretic approach to formulate an efficient path selection.

VII. CONCLUSION

Privacy is an important aspect of communication over web. Mere encryption of messages, tunneling the messages over an encrypted link cannot guard against the malicious attackers. Traffic analysis is a major issue that challenges privacy and measures have to be taken to mitigate them. A number of Anonymous communication protocols have been discussed and analyzed with pros and cons. To maintain anonymity over the web, Onion routing is by far the best solution regardless of performance issues. In onion routing (for wired networks) data is wrapped under multiple layers of encryption and forwarded towards the destination and each node on the route decrypts a layer and forwards it. Tor is the second generation onion routing protocol that outperform by solving the caveats of the conventional onion routing protocol and aims to perform better.

References

1. Chaum David,"Untraceable electronic mail, return addresses and digital pseudonyms, Communications of the ACM 4,2(February 1981
2. Pfitzmann, A. and Hansen, M. 2008. Anonymity, unobservability, and pseudonymity: A consolidated proposal for terminology. Draft
3. Edman, M. and Yener, B. 2009. On anonymity in an electronic society: A survey of anonymous communication systems. ACM Computing Surveys (CSUR), Volume 42 Issue 1, 35 Pages, December 2009.
4. Helmers. A brief history of anon.penet._ - the legendary anonymous remailer., http://www.december.com/cmc/mag/1997/sep/helmers.html.
5. D. Kesdogan J. Egner R. Buschkes "Stop and Go MIXes providing probabilistic security in an Open System", Second International Workshop on Information Hiding, Vol 1523 Portland, Oregon April 1998,pp 83-98
6. Gene TsudikCeki. Gulcu. Mixing E-mail with Babel. In Proceedings of the Network and Distributed Security Symposium - NDSS '96, SanDiego, California pages 216, IEEE, February 1996
7. Paul F. Syverson, David M. Goldschlag, and Michael G. Reed, "Anonymous Connections and Onion Routing ", IEEE Journal on Selected Areas in Communication, Vol 16, No.4, pp 482-494, May 1998, Naval Research Laboratory. 1998
8. ParvathinathanVenkitasubramaniam, VenkatAnantharam, "On the Anonymity of Chaum'sMix", IEEE 2008,Torronto, Canada
9. M. Rennhard, S. Rafaeli, and L. Mathy. " Analysis of an Anonymity Network forWeb Browsing", TIK Technical Report Nr. 129, TIK, ETH Zurich, Zurich, CH, February 2002.
10. X. Fu, B. Graham, R. Bettati, and W. Zhao, "On countermeasures to traffic analysis attacks," in Information Assurance Workshop, 2003. IEEE Systems, Man and Cybernetics Society, pp. 188–195, 18-23 June 2003.
11. Abhishek Mishra ,Parv. Venkitasubtramaniam,"Dummy Rate Analysis of Buffer Constrained ChaumMix", SIGCOMM'11, August 15–19, 2011, Toronto, Ontario, Canada. ACM 978-1-4503-0797-0/11/08.
12. Serjantov, A., Dingledine, R., and Syverson, P. 2002. From a trickle to a flood: Active attackson several mix types. In Proceedings of Information Hiding Workshop (IH 2002), F. Petitcolas,Ed. Springer-Verlag, LNCS 2578.
13. Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for Web Transactions. ACM Transactions on Information and SystemSecurity, 1(1):66–92, November 1998.
14. VitalyShmatikov,"Probabilistic Analysis of anonymity. In Proc. 15th IEEE Computer Security Foundations Workshop, pages 119-128,2002.
15. Stefan Rassy, Raphael Wigoutschnigg, and Peter Schartner, " Doubly-Anonymous Crowds: Using Secret-Sharing to achieve Sender- and Receiver-Anonymity", Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, volume: 2, number: 4, pp. 27-41 August 2011
16. Levine, Brian Neil, and Shields, Clay, "Hordes : A multicast-based protocol for anonymity." Journal of Computer Security, 10(3): pages 213-240, 2002.
17. Paul F. Syverson, David M. Goldschlag, and Michael G. Reed, "Anonymous Connections and Onion Routing ", Naval Research Laboratory. 1998
18. R. Morris M. J. Freedman. 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.
19. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium (August 2004)
20. A. Hintz. Fingerprinting websites using tra_canalysis.In R. Dingledine and P. Syverson, editors, Proceedings of Privacy Enhancing Technologies Workshop (PET 2002), volume 2482 of Lecture Notes in Computer Science, pages 171{178. Springer-Verlag, Apr. 2002.
21. AndriyPanchenko, Lukas Neissen and Andreas Zinnen, "Website Fingerprinting in Onion Routing Based Anonymization Networks", ACM transactions on WPES'11, October 17, 2011, Chicago, Illinois, USA.
22. Ryan Pries, Wei Yu, Steve Graham, and Xinwen Fu, "On Performance Bottleneck of Anonymous Communication Networks",

