I2P Data Communication System

Bassam Zantout and Ramzi A. Haraty
Department of Computer Science and Mathematics
Lebanese American University
Beirut, Lebanon 1102 2801
Email: rharaty@lau.edu.lb

***Abstract*---As communication becomes more and more an integral part of our day to day lives, the need to access information increases as well. Security is currently one of the most important factors to consider in our aim to achieve ubiquitous computing, and with it raises the problem of how to manipulate data while maintaining secrecy and integrity. This paper presents one of the most common widely used data communication systems for avoiding traffic analysis as well as assuring data integrity - I2P. I2P, just like every other technology aimed at securing data, has its pros and cons. The paper presents the benefits and drawbacks of I2P.**

Keywords - I2P; Traffic Analysis; Data Integrity.

I. INTRODUCTION

Since the day the Internet became a common and reliable mechanism for communication and data transfer, security officers and security enthusiasts rallied to enforce security standards on data transported over the globe. The goal was to achieve data integrity and confidentiality, while using a reliable data transport medium, which is the Internet. Whenever a user tries communicating with another recipient on the Internet, vital information is sent over different networks until the information is dropped, intercepted, or normally reaches the recipient. This information identifies where the request is coming from by revealing the user's IP and hence the geographical location, what the user needs from the recipient, and sometimes the identity of the user. The moment the recipient replies back, the same type of information is sent back along with a certain payload (meaningful content) for which the user had requested. Critical information traversing networks is usually encrypted. Sometimes encrypting the payload alone is not enough for users who wish to conceal their identities while communicating with recipients over the Internet. Take for example a reporter working undercover and sending critical information over the Internet to a country that is at war with where the reporter is residing in. If the reporter's identity is revealed, then the reporter might be convicted. Hence, concealing who is sending the information is sometimes much more important than revealing the information itself. In order to conceal the sender's identity, different implementations have proven successful. One of which is the invention of anonymous networks. Anonymous networks go beyond transferring information over the Internet, whereby theoretically, the implementations can be replicated on different communication technologies such as mobile devices, wireless networks, etc. [1, 2, 3]. In 2003, and due to a huge interest as well as considerable advances in P2P concepts, whereby numerous projects for distributed file sharing and P2P networking emerged, a new project called the I2P (*Invisible Internet Project*) was introduced to the public [4, 5, 6]. The main developers and supporters for this project remain anonymous to this date and call themselves nicknames whereby *jrandom* is the main developer and the person responsible for this project, which was later called I2P. *jrandom*, along with many developers studied different anonymous systems at that time (Tor, Tarzan, Freenet, Bitorrent) [7, 8, 9, 10] and then discovered and implemented new and unique ideas for distributed P2P anonymous systems, which promised better anonymity to its users [11, 12].

The paper discusses I2P and presents its benefits and drawbacks that will be discussed thoroughly including the newly introduced methodology. The remainder of this paper is organized as follows: Section two provides the background of I2P. Section three discusses I2P in detail and Section four provides a conclusion.

II. BACKGROUND

I2P is a low latency anonymizing mix network that offers its users a certain level of traffic analysis prevention; hence, hiding the identity of both the sender and receiver, while utilizing a large set of encryption standards to hide data content and to ensure payload delivery. Just like NetCamo [13], I2P is intended to be used with nodes that have I2P system installed. Moreover, just like Tor, I2P is capable of relaying traffic through multiple nodes using tunnels and encapsulated messages of data that are routed until the destination is reached. However, the key difference is that I2P is a message-based system instead of circuit-based as in Tor. Moreover, unlike Tor, I2P is a fully distributed system that does not rely on centralized directory servers to keep track of participating nodes and network performance. Instead, I2P utilizes a modified Kademlia algorithm [14] that handles network and node information that is distributed and maintained among different nodes in the I2P network. After several years of discussions and development, I2P is still considered in the alpha stage whereby the core components and driving engine have been

 ISBN:978-1-61208-113-7

changed frequently and will continue to change due to enhancements. I2P is still not concerned a fully reliable anonymous system, although developers and users can logon to the network for a test drive.

III. I2P AND GARLIC ROUTING

The following sections describe how I2P functions and what makes its corresponding components unique. It is important to note that I2P, while similar to Tor in some of its definitions, differs immensely in its design and implementation.

A. Garlic and Onions, Cells versus Cloves

The Second Generation Onion Routing Project, Tor, as well as the original onion routing design devised a system based on cells whereby a cell is of fixed size that contains encrypted information of either instructions to other onion router nodes, or data/payloads to be delivered to a certain recipient. The onion cell has fixed size in order to conceal hints about information or the content of the data being transmitted from and between nodes in the system. This fixed sized technique was considered as a security and anonymity enforcing mechanism since traffic analysis of fixed size cells could prevent against website fingerprinting, and target/sender correlation attacks. While this technique is indeed effective in high-end Tor nodes where millions of cells are passing every hour through these nodes; however, fixed cells prove inefficient when it comes to end-to-end attacks and time-based attacks since cells are not padded with random data and lack intentionally introduced delays (with latency considerations). Garlic routing was inspired from onion routing whereby garlic cloves are simply a combination of one or two onion cells in addition to extra padded information of random size. Hence, the atomic data unit for the I2P system is indeed the same as the onion system; however, not a single atomic unit is transmitted alone. Instead, previously encrypted onion cells are grouped together, with extra padding, as well as delay/no-delay instructions to other I2P nodes, and then packaged in so called Garlic cloves, which are then passed to other I2P nodes, in an encrypted format. The size of a clove as well as the number of onion cells differs between I2P nodes in order to add additional randomness to the system. As I2P nodes receive encrypted Garlic cloves, they are able to decrypt them (using public and private keys) and then treat each onion cell independently and sometimes with special latency and priority requirements sent by embedded instructions. Each I2P node is then able to repackage received onion cells using new encrypted garlic cloves and then send them to other I2P nodes.

This alone makes I2P a message-based system instead of a circuit-based system as in Tor. However, the notion of circuits and hops still exists. What is important to note is that if two users who have both installed the I2P client software, these users will be able to send information to each other in fully encrypted format, and thus prevent end-to-end attacks to non-global adversaries.

B. Tunnel and Communication amongst I2P Nodes

I2P utilizes a huge set of protocols, encryption standards, and P2P concepts in order to achieve the highest levels of anonymity for its users. This section describes I2P node and user communication in details. However, it is important to stop and visit two vital concepts in P2P networking and to I2P, which are DHT and Kademlia.

C. DHT (Distributed Hashed Tables)

A hash table is a simple algorithm that given a hash function and a certain input, then a *unique* output (depending on the hash function) is derived. Hash functions are extremely efficient in locating values that correspond to a certain input, and the notion of buckets is used to indicate many values a hash function could outputted to when a single input is used. A DHT is a similar concept for decentralized distributed systems whereby one is able to lookup information in a distributed system efficiently. Given two pairs of information *(name, value)* which are stored in a DHT, participating nodes can work together in maintaining and mapping these pair of information amongst each other with minimal amount of resource and network overhead. DHT was crafted after being inspired from inefficient distributed lookup services found in P2P implementations at the time. These P2P implementations where mainly focused at locating resources or files in distributed systems whereby three methodologies were used:

1) Centralized Indexed System: a central system was assigned whereby participating nodes pushed whatever resource listing they had to this system. The central node then performed indexing on this information and any user who wished to locate data present on the distributed nodes queried the central system for the location of data. A similar system that received its 15 minutes of fame was Napster that later faced enormous slowdowns as the size of the files and data increased.
2) Flooded Query System: is another system that required for each query, a user issued, to be distributed to all participating nodes in the system. Although this might reveal the most updated results since no central system performance or update delays might occur; however, allowing such a system to scale was improbable since as the number of search queries increases and as the number of nodes increases, the number of broadcasts and replies also increase. Gnutella is an example of such a P2P system.

 ISBN:978-1-61208-113-7

3) Heuristic Key Based Routing: was utilized by Freenet, whereby a resource was associated with a key and resources with similar keys are located in a cluster or group of similar nodes. So based on the key a user issues, a key based routing is implemented and the query is directed to these set of nodes instead of being broadcasted to all nodes, or a centralized system.

DHT was developed to overcome the above points and is characterized by the following:

1) Decentralization: each node is an independent node that does not rely on a centralized system for coordinating tasks and locating information.
2) Scalability: A DHT system is able to scale highly (millions of nodes) while keeping its phenomenal and efficient search capabilities as is.
3) Fault Tolerance: nodes in a DHT system may join and leave the system while keeping all stored information in the system intact.
4) Performance: Given ***n*** nodes in a system, then as the number of nodes increases or decreases, the system is able to retrieve information in the O(logn) (Big O notation).

DHTs received a great deal of attention by many academic institutes for which implementations like Chord, Kademlia, CAN, Pastry and Tapestry where developed. The implementations differ in some details; however, the overall concept is the same whereby three components are identified: *keyspace*, *keyspace partitioning*, and *overlay network*.

A *keyspace* is a set of sequence of bits of a certain fixed length ***F***. For any content that needs to be stored in a DHT system a filename is hashed and one obtains a hashed value of size ***F*** that corresponds to a certain resource ***R***. The data along with the hashed filename are introduced into the DHT network and the DHT system forwards (through the *overlay network*) such information amongst DHT nodes until the data arrives to the DHT node responsible for keeping track of this file information. A query then given to any node in the DHT system about this filename is hashed and then forwarded (using the *overlay network*) till it arrives to the node responsible for such information.

Nodes in a DHT system are conceptually arranged in circular ring network although physically nodes may be geographically dispersed over the globe. Each node is aware of its successor and predecessor, and traversal of the nodes usually occurs clockwise.

For two keys k1 and k2, *keyspace partitioning* is illustrated as the distance between k1 and k2 represented by $\delta(k1,k2)$, whereby the distance does not relate to network latency or geographical location. Each node in the DHT network is then give a key as its identifier; hence, a node with ID *i* owns all the keys for which *i* is closest to. The *keyspace* is split into contiguous segments whose endpoints are the node IDs, whereby for any two nodes *i1* and *i2* for a key *kx* then *i2* holds all the keys that fall between *i1* and *i2*.

The DHT system may introduce a pool of nodes whereby each of pool is responsible for replicated data content to cater for node joins and departures in the system as displayed in Figure 1.

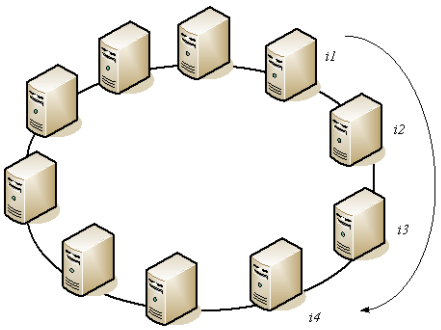

Figure 1. Sample DHT system.

The *overlay network* can be described by the fact that each node keeps track of its adjacent nodes; hence, traversing the system may require an O(n) (Big O notation. Although this might become extremely inefficient for large systems, a series of enhancements where introduced, as in Chord and Kademlia, which reduced the traversal to O(logn) by adding a routing table to each node (through continuous lookups as nodes join and leave the network or as queries traverse the network, nodes are able to keep track changes). The search method using the *overlay network* is executed using a greedy algorithm that forwards/routes searches to the closest node holding a key similar or close enough to the original key. Of course, different DHT designs have different implementations of the *overlay network*.

DHT is considered a core infrastructure that has been used to build many complex systems that have been widely adopted and later modified by many implementations like Bittorrent, I2P, Coral Content Distribution, eMule, and Freenet, and many others.

D. Kademlia

I2P designers have adopted a DHT implementation by Petar Maymounkov and David Mazi`eres, from the University of New York, called Kademlia. This DHT implementation is capable of running in a network where a lot of node joins and departures occur and hence uses a XOR-based metric topology whereby the distance between two nodes is computed as the XOR of the node IDs (knowing that the node IDs are in a certain increasing sequence). Additionally, queries about keys and nodes in a network are recorded by every DHT node through which a query traverses through. This, along with efficient data retrieval (O(logn)) due to a good routing implementation for

locating nodes and data in nodes, makes Kademlia a good candidate for any DHT implementation.

E. I2P Tunnel and Node Characteristics

The I2P network is composed of I2P routers that relay encrypted garlic cloves, and I2P users transmitting and receiving such cloves to each other. I2P routers and destinations (or end-user client nodes) have distinct identification through cryptographic identities, which enables them to send and receive messages as well as form encrypted tunnels. Each I2P node or router in the network has inbound and outbound tunnel(s) established and connected to other I2P gateway(s). The number of tunnels can be increased to form different routes by simply connecting to different I2P gateways. When a message needs to be relayed from a sender to a recipient the message goes through the senders outbound tunnel to the end-point of the tunnel, which is another I2P gateway and that gateway then forwards the message to through a series or hops (or directly) to the gateway of the recipient for which then the message traverses the recipients inbound tunnel and gets decrypted at the recipients node. Senders have no information about the path the message will take except for the gateway the senders have used to release the message. Each I2P router in the I2P network is able to add delays, introduce additional padding, and route information according to a node directory lookup called the *NetDB* (network database based on the Kademlia algorithm). Figure 2 better illustrates the I2P topology.

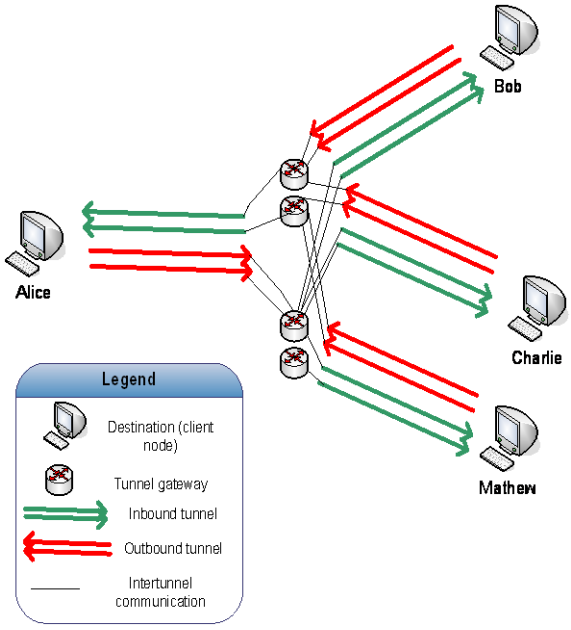

Figure 2. I2P network – A sample of inbound and outbound tunnels used for communication.

In the above illustration, Bob is able to send Alice information in a couple of steps by:

1) Querying the NetDB to retrieve information about routers identity and encryptions keys as well as destination's public keys and reachability of gateways and destinations. This information is stored in the NetDB under two categories: the *RouteInfo* and the *LeaseSet*.
2) Sending messages through its outbound tunnel to a router then in turn converts the stream into the inbound tunnel of Alice.
3) Getting back replies from Alice through Alice's outbound tunnel that in turn gets converted by I2P routers to Bob's inbound tunnels.

As such one can notice that I2P does not have any entry and exit nodes like Tor, and data is encrypted end-to-end among peers in the system. However, in order to achieve this, both sender and recipient need to be on the I2P network connected to at least a single I2P node with two tunnels (inbound and outbound). The amount of tunnel creations is user based. However, the default is creating four tunnels - two for inbound and another two for outbound communication respectively. This could be justified in the event that if one tunnel goes down for any reason, then a secondary tunnel exists. Moreover, adding more tunnels means having different routes to the same destination or a set of destinations. I2P users are also able to choose the number of hops found on the path to a certain destination whereby the illustration shows only a single hop from Bob to Alice. I2P currently supports a maximum of two hops before traffic is routed and delivered to its final destination. Tunnel creations are time based and change every 10 minutes in order avoid any types of attacks on the tunnel encryption. Even if tunnel encryption has been compromised, then payloads - packages in garlic cloves, have already a multilayer of synchronous and asynchronous encryption standards to assure data integrity and anonymity. Another type of tunnel exists, that has not been shown in the preceding graphical illustration, used for I2P node discovery. An explanation about node discovery is discussed in the next section.

Nodes in an I2P network are classified into four different categories according to speed and reliability:

1) **High Capacity**: are nodes that have a higher connection capacities and uptime than the average number of nodes connected to the system.
2) **Fast**: are nodes that are categorized as "high capacity"; however, they are considered with fast connection due to bandwidth (throughput) compared also to the average number of nodes with connection speeds.
3) **Not Failing**: are nodes if it is neither high capacity nor failing.

 ISBN:978-1-61208-113-7

4) **Failing**: are nodes that frequently join and leave the network, or that are queried but found to be always unavailable.

Although there is an additional node classification called "Well Integrated", which means that a node "integration calculation" is above average, there is no explicit information about what that really means. Since I2P is a decentralized and distributed system, with a large number frequent joins and departures of end-user nodes that form the majority of nodes in the system, then there is no way of determining exact node statistics similar to a centralized system as Tor. As a result, node information and categorization is usually kept at every node in the system with the help of Kademlia's node traversal and NetDB data querying. Hence, two nodes that are not well-integrated in the network or with network problems may have varied answers about a particular node in the network.

F. Peer Selection and Tunnel Creation

When a new user wishes to join the I2P anonymous network, the user downloads and installs the I2P software from the official website after performing a checksum on the software itself (to ensure data integrity and that the software has not been tampered with). The I2P software is a Java based client that enables users to look for an available I2P node and connect to that node and therefore join the I2P anonymous system. However, one might wonder how is this done when I2P itself is a distributed and decentralized anonymous network that is hard to keep track of, and connect to, especially when a new node tries to connect for the very first time. As a solution, the I2P developers have introduced a set of hosts' IPs that have a good uptime (*Fast* or *High Capacity*) and which are considered reliable hosts - once the I2P software launches it bootstraps using a randomly selected I2P host IP from the set of preconfigured (*Fast* or *High Capacity*) IPs and logs on the I2P network. If bootstrapping is unsuccessful then the client software can choose another IP until a connection occurs. Once connected, a series of node investigations is carried out by the newly joined node using a second type of tunneling used only to traverse the I2P network looking for available nodes. Node traversal begins taking place gradually as the new node starts building tunnels with random nodes in the system (Kademlia DHT algorithm). At every tunnel creation the new node will query existing I2P nodes for available nodes for which it can connect to, and since tunnels do not usually last long, the rate of discovery becomes high and the list becomes larger with a considerable uptime.

NetDB stores two sets of data: *RouterInfo* and *LeaseSets*. *RouterInfo* gives users or nodes in the system the set of information to contact a specific router in the network. *LeaseSets* gives nodes in the system information for contacting a particular destination node where data will be delivered, also composed of destination tunnel address(s), public keys, and the destination's tunnel uptime for path reliability. While *RouterInfo* information may not change frequently since users connect to dedicated gateways, *LeaseSets* for nodes do change frequently since users often change tunnels every 10 minutes and hence reachability is changed too. End user nodes usually choose first *Fast* or *High Capacity* I2P nodes so that tunnel creation is reliable for sending and receiving data, then the nodes establish another set of inbound and outbound tunnels called the *exploratory tunnels* with less capacity (*Not Failing*) nodes to query the network about other available nodes and to obtain the NetDB information carrying encryption specifications about how to connect and how to reach other nodes.

G. System Encryptions Standards for Communication

I2P uses four different types of cryptographic algorithms for ensuring communication reliability, anonymity, and data integrity during data transmission through multiple paths. As a result, symmetric, asymmetric, signing and hashing algorithms have been used all together to strengthen the security on communication. Any established tunnel uses 2048 bit ElGamal/Session Tags with 256 bit AES in CBC mode encryption. Signatures use a 1024bit DSA algorithm with a 320 bit seed. TCP connections operate using a 2048 Diffie-Hellman implementation at the moment. Jrandom [5] contains an illustration of the components used in a single tunnel, which happens to be Alice's outbound tunnel and Bob's inbound tunnel for one way communication where Alice is sending a message to Bob.

H. Exit Policy for Internet Communication

I2P can only assure end-to-end privacy and integrity if the two peers involved in communication have joined the I2P network. I2P was never meant to be used for Internet communication. However, some nodes in the network are running exit proxies that enable users to reach destinations located on the Internet.

Similar to Tor, end-to-end encryption with nodes outside the anonymous system are released from the exit nodes without any encryption to reach the destination. Once a reply from the destination sent back to the original sender, the exit node will re-encrypt and repackage garlic cloves to be sent back to the sender. I2P is not meant for anonymous Internet browsing, but for anonymous P2P communication whereby end-users are able to send and receive data anonymously. I2P also supports services similar to Tor's *hidden services* to users in the system. An example of a *hidden service* is websites that are posted anonymously, whereby their IP and ID are not revealed to visitors. Websites are given the extension (**.i2p)*. Although *hidden services* are not purely the focus of this research; however, a few remarks are commented on in the critique section for I2P.

 ISBN:978-1-61208-113-7

H. Threat model

I2P protects against a number of attacks such as: Brute force attacks, Timing attacks, Intersection attacks, Denial of service attacks, Tagging attacks, Partitioning attacks, Predecessor attacks, Harvesting attacks, Sybil attacks, Cryptographic attacks, Development attacks, and Implementation attacks.

I. Critique

I2P is by far the most complicated and most promising anonymous P2P system for many aforementioned reasons. I2P is the product of years of continuous development by a number of dedicated developers that have conducted enough research on existing anonymous systems to come up with a new decentralized system that offers better anonymity to users. Nevertheless, I2P was never meant to be used outside the participating nodes in the system itself. Hence, users connecting to I2P and wishing to browse the Internet or carry out other Internet tasks like chatting, sending emails, and talking using VoIP are not advised to do so. The reason behind this is simply because I2P was never designed for communication between an anonymous system and the Internet. Although there have been some assigned nodes with gateways to the Internet, the nodes are limited in number and are well-known to I2P users.

Just like any anonymous system, I2P has its strengths and weaknesses. The advantages of I2P are:

1) **Message-based instead of Circuit-based:** I2P is a message-based system whereby data packets generated be senders and receivers are encrypted and then wrapped randomly together (using the garlic protocol) and then sent across the I2P network whereby cloves bounce through random hops before reaching their final destination. In comparison with circuit-based, messages no longer need wait for a peer to establish a tunnel through other peers before proceeding with data delivery. Messages simply traverse pre-created tunnels through available nodes on the system. This immensely reduces the overhead of creating tunnels and adds randomness in the system while allowing hops (or router nodes) to control data delivery and add variable latency as well as padding to messages.
2) **Various Protocols Support:** While other anonymous systems restrict Internet communication to a number of protocols like HTTP, and hidden publishing services, I2P offers a wide range of internal services like P2P services (BitTorrent clone), anonymous hidden services like web publishing, anonymous SMTP, file sharing, and anonymous chatting. These protocols have been developed by enthusiasts as plug-ins on top the I2P system that enable a user logged in to I2P to communicate with other users anonymously.
3) **New P2P Infrastructure over the Internet:** The previous point may give an insight on the future of P2P file sharing and communication on the Internet. During these times, the P2P file sharing activities on the Internet have been criticized for housing and transferring illegal and copyrighted content amongst peers in the community. BitTorrent, eMule, Gnutella, and many others are now considered gateways to tremendous amounts of illegal (and legal) content that is being tracked, along with P2P participants, by copyright entities such as the RIAA, DMCA, Interpol, and other similar organizations. Anirban Banerjee, Michali Faloutsos, Laxmi N. Bhuyan's study [15] of P2P file sharing and communication tracking, has shown that file sharing communities now form *block-nets* in order to block certain known legal entities that have been identified using various methods. *Block-nets* are composed of a list of suspected IPs, circulated amongst different P2P systems and regularly updated, for which users trying to communicate from this list of IPs are blocked from participating and joining P2P systems due to their activities in locating and tracking P2P end users. There have been some documented incidents where a number of users have been tracked down and sued over a number of copyright violation using commonly used P2P systems. By introducing I2P as an infrastructure for P2P protocols for anonymous communication, P2P activities would therefore become anonymous to all participating parties. With header and payload encryption, as well as resource hiding, multi-packet routing, and content distribution, entities wishing to track communication will be find it difficult to do so and will be spotted and hence blocked (based on *block-nets*). An example of such a system is *Freenet* [9]. The aim of course is not to allow illegal content to be freely distributed over the Internet; however, to anonymize communication for users on the system. This idea is not bullet proof, as P2P trackers may login to the system anonymously using another set of IPs and conduct various types of attacks, irrelative of time and cost, to pinpoint and breach the system, if need be.
4) **Open Design, Open Source Code:** The developers of I2P, led by *jrandom*, have kept their identities anonymous for many reasons one of which to indicate that the system is anonymous not only in its functionality but in its development also. The system has an open design whereby as enthusiast may login to the I2P website and check the recent and previous developments of I2P and participate in forums while remaining anonymous. The Java

 ISBN:978-1-61208-113-7

source code for the I2P client and server software are freely available for users to inspect and audit if need be. This adds more trust in the system and for adopting the system especially that the system is not sponsored by any government agency as in Tor.

5) **Different Encryption Techniques:** I2P employs a good set of algorithms ranging from symmetric, asymmetric, singing and hashing algorithms that have been used primarily to hide the identity of users in the system and to ensure the integrity of data delivered to peers along the communications paths. With 2048 bit encryptions keys and newly introduced session tags to identify communication amongst peers, as well as defend against replays, I2P stands out to be one of the few anonymous system encompassing such a large number of encryption technologies.
6) **Distributed and Decentralized System:** By making use of a DHT implementation based on Kademlia, as well as removing centralized entities for managing the nodes on the network, the I2P system is protected against attacks on its directory servers. The I2P system is a self-healing and self-organizing anonymous system that is able to keep track of nodes in the system while part of the network is under attack. Moreover, attacking random/specific nodes in the system appears useless as there is no single entity handling I2P system management. Additionally, if *global adversaries* are able to block known I2P nodes for users to connect to, then it will only require another (unknown to *global adversaries*) user node to join the I2P network and then announce, through different means, that it was capable of joining the network and that node itself will become a gateway for new nodes wishing to join the I2P network.
7) **Different Types of Unidirectional Tunnels:** Almost every type of anonymous system that was designed and implemented uses a single tunnel for moving data back and forth from sender to receiver, and hence encrypted tunnels where multi-directional carrying not only encrypted data payloads, but also instructions to other nodes in the system. I2P designers have decided to separate and segregate the role of tunnels by introducing a new set of tunnels such as one for sending data (along with some instructions), one for receiving data (along with some instructions), and another for exploring the network. The latter uses nodes with less bandwidth on less reliable/slower connection nodes. The significance of this segregation is to enhance the amount of peers participating in communication and therefore increase the number of hops along the path of data being transmitted and received. This modification adds a minimum of twice the number of participating nodes in any communication as compared to Tor, which utilizes a single tunnel for sending and receiving data streams as well as instructions to other nodes.
8) **End User Node Participation in Communication:** The new design for I2P encourages that every node joining the I2P needs to use part of its bandwidth as a relay node and pass data to other peers in the system. This approach not only makes use of the end users, who are usually the majority of the nodes in the system using the system, but also adds more hops to any communications that allows more randomness when choosing hops.

The disadvantages of I2P are:

1) **Vulnerability to Partitioning Attacks:** Since I2P utilizes Kademlia for maintaining the distributed system and keeping nodes in contact (using NetDB). Kademlia is susceptible to partitioning attacks that may disconnect targets in the system and therefore reveal the identities of all parties involved in a communication stream. A partitioning attack is an attack that aims at directing end users in the system to connect to a smaller set of malicious nodes only (smaller relative to the complete set of nodes in the system), whereby the malicious nodes are able to simulate the functionality of the anonymous system to the target node, and whereby a user is still able to establish a number of tunnels and select multiple hops. However, all identities for the sender and receiver are actually compromised along the pathways as the nodes participating in communication are malicious nodes. Strong adversaries are also capable of deliberately blocking certain destinations. Hence, other legitimate nodes in the system; and therefore, disconnecting the target at once from the rest of the nodes in the system and then introducing malicious alternative nodes with another set of *NetDB* options and routes. This attack coupled with other types of attacks such as Sybil and Time attacks may fully exploit the identities of the senders and receivers in the system, as well as the data content, especially if one of the malicious nodes is used as an exit node to the Internet.
2) **Possible Intersection Attacks:** An intersection attack is an attack that monitors a certain target and then watches the amount of nodes present and connected to the system constantly. Due to tunnel rotation and variation in target reachability, the attacker may eliminate nodes that have not participated in communication with the target until the target's paths are narrowed down. This will also leave a set of nodes that form these paths

 ISBN:978-1-61208-113-7

exposed for monitoring - seeing as a message traverses the paths from source to destination and vice versa the node is detected. Intersection attacks are strengthened immensely when coupled with other types of attacks such as Sybil and Timing attacks.

3) **Lack of Node and Bandwidth Monitoring:** I2P is a decentralized and distributed system that does not keep track of overall network bandwidth usage and monitoring except for client and router nodes that self-monitor themselves. I2P nodes are capable of storing and graphing their own connection as well as downloading *NetDB* information that are offered by other nodes in the system. One might question the possibility for having an overview of system performance, network bottlenecks on certain nodes, and total bandwidth as well as the number of participating entities in an I2P network (or a graph representation of the connected nodes). Some I2P security enthusiasts argue that revealing such information might risk anonymity as well as reveal weak points in the system. Moreover, as certain peers and routers in the network can variably change their relay capabilities (bandwidth options offered to other peers in the system for relaying their traffic), and as random joins and departures of nodes occur, it might be tough to graph the network or reveal overall relay information without having a centralized polling system to keep track of such frequent change in information. Given that I2P is a decentralized system then the possibility becomes hardly possible, and hence per-node resource and network performance is kept in each I2P node's NetDB information that have decided to participate in the system. On the other hand, when a wide distributed attack is carried out on most of the nodes, and when nodes become unreachable, then would one (or the system) analyze this type of misbehavior and consequently categorize the incident as a false positive, false negative, a single or multiple nodes experiencing connection problems or failure, a DoS attack, a partitioning attack, or any type of network related attack? How would developers in the system realize that router nodes have been flooded with tunnel connections by a large number of peers in a resource consumption attack, whereby fake packet generation (gradually to camouflage the attack and can usually span days and even weeks) can flood the entire network and hence leave the attack to appear as normal network congestion due to a large number of joining parties while in fact it would be a variant of a Sybil Attack?

4) **NetDB Conflicts and Resolution:** The previous point mentions that NetDB stores node information relayed to different clients and routers in the system whereby it contains information about tunnel, node reachability, and encryption information. Now consider the case where a client node connects to the I2P network. During its network investigation for available nodes.

5) DoS Attacks: The I2P team identified three types of attacks that the system can suffer from, and for which the solutions are questionable. The following briefly explains the attacks:

a. Greedy User Attacks: This is actually is not a form of malicious attack on the system, but more associated with a depletion of available relaying bandwidth.
b. Starvation Attacks: The attack is similar to a Sybil Attack whereby nodes joining the I2P network offer connections to other non-malicious nodes in the system. However, after tunnel creation the malicious nodes drop all incoming and outgoing packets to the newly connected nodes. This will cause the nodes to experience frequent network failures as they will not be able to send and receive data using these tunnels. Additionally, this will cause more tunnels to be established with other non-malicious I2P routers due to the lack of connectivity with current malicious nodes.
c. Flooding Attacks: Is an attack that allows a malicious user to introduce a node or a set of node that inject huge amounts of meaningless/meaningful traffic with destinations to inbound and outbound nodes of different peers in the system. Similar to a network DoS attack, nodes in the I2P network receiving such traffic can do nothing to stop this traffic since any node on any network cannot control the amount of traffic it is receiving. I2P developers argue that if nodes detect that huge amounts of traffic are detected, then they can disconnect their tunnels and reestablish new tunnels with other nodes.

IV. CONCLUSION

This paper presented the I2P anonymous-related systems, and their corresponding details that have made them such a success. The paper also commented on the pros and cons of I2P's implementation.

Avoiding traffic analysis and hiding the identities of users is the aim of any anonymous system. However, since most anonymous systems rely on aging encryption technologies for which global adversaries are a capable of compromising, then the integrity of data might be at stake.

This paper also introduced vital topics that need to be further researched such as creating virtual interfaces for

 ISBN:978-1-61208-113-7

making all types of traffic to traverse anonymous systems, as opposed to socket proxies, in order to maximize securing the identity of users on the system and support the widest types of applications possible. Such virtual interfaces can exist in order to ease selecting which types of traffic can pass through the anonymous system and which can be bypassed to leave the anonymous system's utilization at optimum levels.

One of the key elements that worry anonymous systems researchers is QoS for the bandwidth utilized by peers on the systems and the overall network performance. Although this has been slightly commented on, more research in QoS and a bandwidth-choking approach is required while concentrating on security and functionality implications.

In the near future, we plan to focus our work on avoiding traffic analysis and at the same time assuring data integrity using a quorum-based approach. We plan to introduce this work to different anonymous systems researchers and communities for a possible implementation and real testing on existing systems.

ACKNOWLEDGEMENT

This work was funded by the Lebanese American University.

REFERENCES

[1] D. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," Communications of the ACM, vol. 24, no. 2, Feb. 1981, pp.84-88.

[2] R. Dingledine, M. J. Freedman, and D. Molnar, "Peer-To-Peer: Harnessing the Power of Disruptive Technologies," Free Haven Project. Retrieved on December 14, 2010 from http://freehaven.net.

[3] O. Sandberg, "Distributed Routing in Small-World Networks," Retrieved on February 10, 2006 from http://torr.eff.net.

[4] The Anonymizer. Frequently Asked Questions, Retrieved June 14, 2010, from http://anonymizer.com.

[5] J. Jrandom, "I2P Anonymous Network: Technical Introduction," Retrieved on December 13, 2010, from http://www.i2p2.de/techintro.html.

[6] J. Jrandom, "Introducing I2P: A Scalable Framework for Anonymous Communication," Retrieved December 13, 2010, from http://www.i2p.com.

[7] R. Dingledine and N. Mathewson, "Tor Path Specification," Retrieved on December 13, 2010 from http://torr.eff.net.

[8] M. Freedman and R. Morris, "Tarzan: A Peer-to-Peer Anonymizing Network Layer," Retrieved on December 14, 2010, from http://www.cs.princeton.edu/~mfreed/docs/tarzan-ccs02.pdf.

[9] I. Clarke, S. Miller, T. Hong, O. Sandberg and B. Wiley, "Protecting Free Expression Online with Freenet," IEEE Internet Computing, Retrieved on December 13, 2010 from http://torr.eff.net.

[10] R. Thommes, and M. Coates, "BitTorrent Fairness: Analysis and Improvements," Retrieved on December 13, 2010, from http://www.tsp.ece.mcgill.ca/Networks/projects/pdf/thommes_WITSP05.pdf.

[11] D. Goldschlag, M. Reed and P. Syverson, "Hiding Routing Information," Retrieved on December 13, 2010, from http://stinet.dtic.mil/cgi-bin/GetTRDoc?AD=ADA465075&Location=U2&doc=GetTRDoc.pdf.

[12] S. Katti, J. Cohen and D. Katabi, "Information Slicing: Anonymity Using Unreliable Overlays," Retrieved on December 13, 2010, from http://nms.lcs.mit.edu/~dina/pub/slicing-nsdi.pdf.

[13] Y. Guan, X. Fu, D. Xuan, P. Shenoy, R. Battati, and W. Zhao, "NetCamo: Camouflaging Network Traffic for QoS-Guaranteed Mission Critical Applications," Retrieved on December 13, 2010, from http://ieeexplore.ieee.org/iel5/3468/20237/00935042.pdf.

[14] P. Maymounkov and D. Mazieres, "Kademlia: A Peer-to-Peer Information System based on XOR Metric," Proceedings of the First International Workshop on Peer-to-Peer Systems - IPTPS. March 2002.

[15] A. Banerjee, M. Faloutsos and N. Laxmi, "P2P: Is Big Brother Watching You?," Retrieved on December 13, 2010 from http://www1.cs.ucr.edu/store/techreports/UCR-CS-2006-06201.pdf.

 ISBN:978-1-61208-113-7