How to Find Hidden Users: A Survey of Attacks on Anonymity Networks

Esra Erdin, Chris Zachor, and Mehmet Hadi Gunes

***Abstract*—Communication privacy has been a growing concern, particularly with the Internet becoming a major hub of our daily interactions. Revelations of government tracking and corporate profiling have resulted in increasing interest in anonymous communication mechanisms. Several systems have been developed with the aim of preserving communication privacy via unlinkability within a public network environment such as Tor and I2P. As the anonymity networks cannot guarantee perfect anonymity, it is important for users to understand the risks they might face when utilizing such technologies. In this paper, we discuss potential attacks on the anonymity networks that can compromise user identities and communication links. We also summarize protection mechanisms against such attacks. Many attacks against anonymity networks are well studied, and most of the modern systems have built-in mechanisms to prevent these attacks. Additionally, some of the attacks require considerable resources to be effective and hence are very unlikely to succeed against modern anonymity networks.**

***Index Terms*—Anonymity, anonymous communication, privacy, onion routing.**

I. Introduction

PRIVACY has been an important concern with the rapid expansion of the Internet in our daily interactions. When we visit a website or send an email, our client sends data packets through several routers/servers that contain information to identify the sender and the receiver of the message. Even if the data within the packet is encrypted, the IP header is visible and reveals the communicating parties. By observing communications through a link or router, third parties can track interactions between users based on the source and destination IP addresses. Subsequently, adversaries can obtain considerable information about the interaction pattern between the sender and the receiver.

Anonymizer mechanisms help users to keep their activities private via unlinkability and unobservability. Researchers design anonymity networks that build an overlay network among volunteer systems on the Internet. By using anonymizer networks, users can communicate with one another without revealing their identities or locations. Since the milestone paper on anonymous communications [1], anonymity research has expanded to many areas such as anonymous emails [2], anonymous voting [3]–[5], communication censorship [6], [7], private information retrieval [8]–[10], taxonomy [11], traffic analysis [12]–[14], and others [15], [16]. Anonymity services are provided by either commercial companies, e.g., `Anonymizer.com`} and `GoTrusted.com`, or open source developers, e.g., Tor [17] and Invisible Internet Project (I2P) [18].

Practical situations, such as freedom of speech and anti-censorship, caused a growing interest in anonymous communication through the Internet. Some organizations, including government and military, may want certain communications to remain hidden as the exposure of communication link between both parties may be undesirable or harmful. For instance, human rights groups have found anonymity networks valuable for communicating with the outside world under the watchful eye of an oppressive regime. Citizens of these countries also express their thoughts and ideas through anonymity networks [19]. Similarly, law enforcement and security researchers may use anonymity networks while investigating malicious online activities [20]. Leaving a government or business IP address in the logs of a system under investigation can tip the bad guy off or leave the investigator vulnerable for retaliation. Users may even want to browse the web without being tracked about their personal browsing habits.

Along with the good uses of anonymity networks, there will always be those who use these technologies to hide their malicious activity. For instance, a student, whose identity was unknown at the time of his threats, used an anonymously created `myspace` account to repeatedly send bomb threats to his school, thus, disrupting classes [21]. It was later revealed through the affidavit for a search warrant that the FBI has a Computer and Internet Protocol Address Verifier (CIPAV) tool that acts much like malware. The tool could collect information to fingerprint targeted systems despite the operator's use of an anonymizing technology. While a law enforcement official might use them to solve a crime, a malicious person or an oppressive country might use these attacks on victims. Understanding such de-anonymizing attacks is paramount to users of these technologies as they must understand the limits of the technology they rely on.

Even though there are surveys of the anonymous communication networks [22]–[24], there is not a comprehensive survey of attack mechanisms on anonymity. The purpose of this study is to provide an overview of the potential attacks on the anonymity networks and examine the risks and protection mechanisms for users of the these networks. To this end, we first overview currently deployed anonymity technologies including Tor and I2P [25], [26]. Then, we survey the de-anonymization

Manuscript received November 3, 2014; revised May 6, 2015; accepted June 19, 2015. Date of publication July 8, 2015; date of current version November 18, 2015. This work was supported in part by National Science Foundation Award 1301726. (*Corresponding author: Mehmet Hadi Gunes*.)

The authors are with the Department of Computer Science and Engineering, University of Nevada, Reno, NV 89503 USA (e-mail: eerdin@cse.unr.edu; czachor@cse.unr.edu; mgunes@unr.edu).

Digital Object Identifier 10.1109/COMST.2015.2453434

approaches so that we can better understand vulnerabilities in the anonymity technologies. We categorized the attacks into two, i.e., *application based attacks* and *network level attacks*. Application based attacks are often possible because of either the carelessness of the user or the application developer. First involves the user to reveal identifying information while communicating anonymously. There is not much the technology can do to prevent such compromises as the user should be diligent in their transmissions when they want to be anonymous. The latter is an issue with the implementation of applications that might leak identifying information. Network level attacks on anonymous communication are often out of the control of the user. The attack generally exists because of either a limitation of the anonymity network or a trade off that was made in the development of the anonymity network. For both categories, the effectiveness of attack approaches rely on certain conditions and hence their success vary depending on the state of the attacked anonymity network. Moreover, while some attacks are possible in theory, their effective deployment in real life is prohibitive.

In the rest of this paper, Section II presents the preliminaries on anonymous communication. Section III overviews application based attacks and Section IV discusses network level attacks. Section V provides a discussion of the de-anonymizer techniques. Finally, Section VI concludes the paper.

II. Background

In this section, we provide definitions of commonly used terms and a brief overview of currently utilized anonymity networks.

A. Definitions

A communication network is composed of *messages*, *senders* and *recipients*. Sender and recipient can be a client, server or peer in the communication network. As they exchange messages, a malicious entity may monitor the traffic transmitted between the sender and recipient. In order to better understand anonymous communication systems, it is essential to clarify the terms *anonymity*, *unlinkability*, and *unobservability* [23], [27].

Anonymity: Anonymity is defined as the state of being not identifiable within a set of subjects, called the anonymity set. All subjects that can cause an action constitute the *anonymity set*. In other words, a message can possibly be sent by any of the senders who belong to the sender anonymity set and, similarly, a message can be received by any recipient who might be anonymous within the recipient anonymity set.

Unlinkability: A subject of the system can make (multiple) use of a resource without an adversary being able to link the use to a particular subject. That is, an adversary observing the senders and recipients of a network is unable to identify the communicating parties, and the ability of understanding the communication between individual parties does not increase by observing the network.

The relation between anonymity and unlinkability is that anonymity of a communication in a system is not linkable to any particular identity and the anonymity of a particular identity is not linkable to any communication. That is, any particular message sent through the network is not linkable to a particular sender in the sender set which ensures sender anonymity.

Unobservability: Observers cannot observe the subjects of interest from any other subjects in the communication network. Similar to anonymity sets, unobservability of subjects are defined with respect to the *unobservability sets*. Sender unobservability assures that it is not noticeable whether any sender within the unobservability set sends the message. Similarly, recipient unobservability assures that it is not noticeable whether any recipient within the unobservability set receives the message.

B. Current Anonymity Networks

Anonymous communication designs can be categorized based on the network type, transmission protocol, anonymity properties, or adversary capabilities. In general, anonymity networks are divided into two categories: *high-latency systems* that incur high delay overhead for the message to travel through the network and reach its destination, usually ranging from hours to several days and *low-latency systems* that has minimal delay overhead for the message to reach its destination, i.e., within seconds [22], [24]. High-latency anonymity systems often provide stronger anonymity and mostly used for non-interactive applications that can tolerate higher delays. On the other hand, low-latency systems minimize the delay overhead and are suitable for real-time applications such as instant messaging and web browsing.

Even though several anonymity networks have been developed for anonymizing user communications, currently Tor and I2P are popular. As non-popular systems cannot provide large enough crowd to provide anonymity [25], [26], we do not include them in our discussion. Although Tor and I2P share a few similar mechanisms to provide user anonymity on the Internet, they differ in their original goal. While Tor provides anonymity for users to access the current systems on the Internet via an overlay network, I2P builds an anonymous overlay ecosystem on top of the Internet. Moreover, these networks run in different layers; Tor uses TCP protocol (i.e., transport layer) whereas I2P uses IP protocol (i.e., network layer).

Onion routing is the most prevalent design for low-latency anonymous communication [28], [29]. The structure of these systems is composed of layer upon layer of encryption which encapsulates the message to be transmitted through couple of relays. Each relay has a public and a private key. The public keys are known by all users so that they can establish a communication path. A client first constructs an encrypted tunnel through the network by using public-key cryptography. After setting up the circuit, symmetric key cryptography is used to transfer the data. Both Tor [30], and I2P [18] utilize the onion routing approach (I2P is called garlic routing) as it has separate keys for outbound and inbound tunnels.

Tor is a low-latency, overlay network designed to protect a user's identity when communicating with various applications through TCP [17], [30]. While using Tor, a user can browse the web (or use other TCP-based services) without leaving a trace of his/her IP address in the logs of the various servers

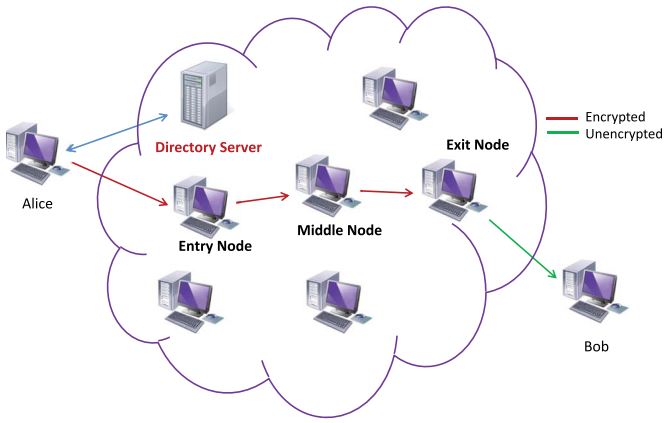

Fig. 1. A sample Tor circuit between two users.

s/he visits. This is accomplished through the use of a user level application referred to as an Onion Proxy. The Onion Proxy establishes a local SOCKS proxy on the user's system which can be connected to by other applications on the system. User data is sent to the SOCKS proxy while the Onion Proxy sends the data through the Tor network.

The Tor network consists of volunteer run systems referred to as Onion Routers. Designated Directory Servers provide a list of systems running the Onion Router software along with their IP address and the public key. The directory contains the router descriptor of each listed onion router and a status document containing measured bandwidths of onion routers. Only onion routers that are verified via their identity keys are listed in the directory. There are multiple directory servers to protect against active attacks against the directory servers. The Onion Proxy starts by connecting to a directory server and collects a list of online routers. The Onion Proxy incrementally builds circuits through three or more randomly chosen Onion Routers as seen in Fig. 1. The final relay, i.e., exit relay, in the circuit will communicate with the destination on behalf of the user. Each relay only knows its direct successor and predecessor to prevent the identification of the source and destination by a single system, and hence provides unlinkability.

Tor clients use a path selection algorithm to improve performance and prevent choosing a corrupted Onion Router as an entry guard while building circuits. Certain Onion Routers are marked as entry guard and exit relay. A Tor relay can be selected as an entry node only after it has been established to be fast, reliable and stable. A relay goes through two phases in 8 days until it can be named 'reliable' to prevent resource exaggeration and 'stable' to enhance communication success [31]. Exit relays are also selected from the Onion Routers with matching exit policies.

The network status and router descriptors maintained by the directory servers are fetched by the Onion Proxy. The router descriptor contains a self-advertised bandwidth and the network status document contains measurement by the directory servers. As long as the measured value is available, it will be preferred due to the fact that self-advertised information might not be trustworthy. The bandwidth information is used to select the intermediate and exit routers in a weighted probabilistic manner where a router with a higher bandwidth is more likely to

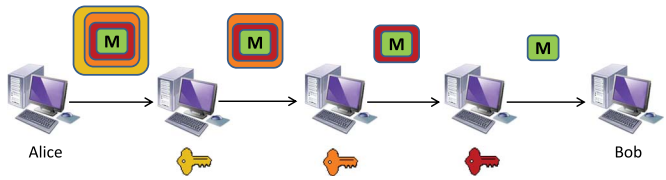

Fig. 2. Tor Onion routing.

be chosen [32]. The Onion Proxy maintains a list of potential entry guards, chosen from the list of all Onion Routers with a long uptime and known to be fast and stable. Onion Routers serving as exit routers can also be considered as entry guards and intermediate routers only if they have sufficient capability [32].

To prevent identification by a malicious entity running multiple relays, the traffic is encrypted in layers as seen in Fig. 2. Encrypted communication is achieved through the use of both asymmetric encryption with public keys and symmetric encryption with session keys. During the circuit building phase, the source provides a symmetric key to each of the relays in the path. These keys are securely exchanged using the asymmetric encryption. Thus, no router in the circuit knows the symmetric key that the user shares with another relay in the circuit. The message to be sent is encrypted with each shared symmetric key starting with the exit relay's key and ending with the entry relay's key. This provides a message with multiple layers of encryption, i.e., the onion. When the message arrives at a router, it decrypts the outermost layer and forwards it on to the next router until the message reaches the destination.

Packets are sent in the reverse direction based on the labels that relays store for each circuit. Any response sent from the destination to the exit relay are then encrypted in the reverse order. At each router between the original destination and the initial sender, a layer of encryption is added until the message arrives at the original sender. Because the original sender knows each key, it can decrypt the message. These encryption layers ensure sender unlinkability as the complete path is only known to the sender and hence a global observer will be unable to stitch observed links to obtain complete path.

Tor can also provide anonymity to recipients. Servers configured to receive inbound connections only through Tor are called hidden services. The aim of these hidden services is to make it possible for users to hide their locations while offering services like web publishing. Instead of revealing a server's IP address, a hidden service is accessed through its onion address. By using Tor rendezvous points, Tor users can connect to these hidden servers, each party without knowing other's real identity.

Invisible Internet Project (I2P) is an overlay network offering low-latency anonymization services that identity-sensitive applications can use [18]. While I2P shares some similarities with Tor, it is different in its goal. I2P, essentially, provides an anonymous "ecosystem" within the Internet to have users access content provided by other users of the anonymity network. While initially Tor focused on the sender anonymity, I2P focused on both the sender and the receiver anonymity. I2P is designed to interact with applications specifically designed for I2P network and uses outproxies to browse the network outside the I2P overlay.

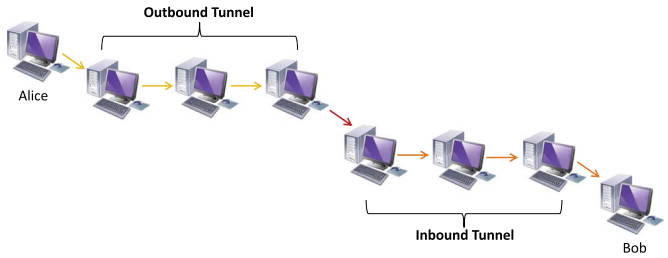

Fig. 3. I2P communication tunnels.

The I2P network is formed by peers (nodes/routers) running the I2P software, which allows applications to communicate through the I2P overlay network [33]. The core part of the software is the I2P router, which is responsible for maintaining peer statistics required for peer selection, performing cryptographic operations, building tunnels, providing services, and relaying messages.

Super-peers, called floodfill peers, are used to build and manage a network database, i.e., *netDB*, based on a distributed hash table. Each floodfill peer is only responsible for information of a specific part of the network. Peers with sufficient bandwidth can be promoted to floodfill peers if the number of available floodfill peers drops below a certain threshold [34]. The netDB stores two types of data, a routerInfo structure that describes an I2P peer and a leaseSet for each known service. All I2P peers are identified by routerInfo, which contains important information about the peer, e.g., IP address, port, peer ID, I2P stable version number, public key, and a hash-identifier. LeaseSet is used to store information about how to contact an internal I2P service, i.e., the destination to be communicated with.

I2P relies on inbound and outbound tunnels to provide unlinkable communication. A new tunnel is established by selecting a set of I2P peers [24]. Tier-based peer selection determines the peers that will be used to build the tunnel. Peer profiling categorizes peers into tiers where peers with similar performance characteristics belong to the same tier [34]. Every 30 seconds all profiles are sorted into three tiers based on various metrics such as bandwidth and speed. Peers that claim to know many of the other peers are grouped as well-integrated tier, peers that are known to most likely accept tunnel build request are grouped in high-capacity tier, and peers from the high-capacity tier with high bandwidth are grouped in fast tier. All other peers are placed in not-failing tier.

If Alice wanted to protect her identity while sending information, she would use an outbound tunnel as seen in Fig. 3. Fast tier peers are the first ones used to build a tunnel. If there is not enough fast tier peers, high-capacity peers are selected. To communicate with an I2P service, the router first needs to retrieve the destination of this service from a floodfill peer. Likewise, if Bob, i.e., the destination, wants to protect his identity while receiving information, he would use an inbound tunnel, i.e., a pseudonym. These tunnels can be viewed similar to the circuits used in Tor. I2P uses *Garlic routing*, which sends garlic messages that can contain multiple so-called cloves, i.e., data messages with additional routing instructions. The actual data messages are end-to-end encrypted with the receiver's public key. The garlic message itself is encrypted multiple times using the symmetric keys negotiated with the tunnel peers. Alice's outbound tunnel consists of three relays in the network using layered encryption to ensure each relay only knows its predecessor and successor. All messages Alice sends will be routed through this outbound tunnel to provide her sender unlinkability. When traversing the tunnel, each peer removes one layer of encryption until the garlic message reaches the outbound tunnel endpoint. On the other end of the communication, Bob has an inbound tunnel providing a pseudonym to provide him receiver unlinkability. Inbound tunnel also consists of three relays. Once inbound tunnel is established, Bob will inform the I2P users that s/he can be reached through the starting point of the inbound tunnel (also known as the inbound gateway) and this pseudonym will be stored in a distributed network database. The outbound endpoint forwards each message to its destination's inbound tunnel gateway. The inbound gateway will forward the garlic message to the actual recipient, i.e., Bob, while each peer participating in the tunnel adds a layer of encryption using the symmetric keys. Only Bob is able to remove all encryption layers of the garlic message as well as the end-to-end encryption of the data-messages.

When Alice and Bob want to communicate, they will check the network database to find the location of other user's inbound gateway without actually knowing the real identities or actual locations. A key exchange will take place between Bob and Alice using Diffie-Hellman Station-to-Station protocol to protect data while traveling between the outbound and inbound tunnels.

There are certain differences between Tor and I2P networks [24]. The main difference between both is the focus of the networks. While Tor was designed with the intention to enable anonymous internet browsing, I2P's focus was to provide an anonymous overlay network isolated inside the Internet. It is, however, possible to utilize I2P outproxies to reach the Internet and enable anonymous web browsing. Another difference with the Tor network is that I2P tunnels can only be used unidirectional compared to the Tor's bidirectional tunnels. Although, both Tor and I2P are based on layered encryption, garlic encryption used by I2P offers the possibility to store multiple messages inside the innermost layer. Another difference between these two networks is that the SOCKS interface that Tor uses is only able to transmit messages over TCP, while I2P has the choice between UDP and TCP. Both Tor and I2P run specific node selection algorithms to improve end-to-end performance. Tor distinguishes between entry and exit nodes while I2P does not have such classification. Each peer in I2P can be selected either the first, second or the third peer in the tunnel. To be selected as relays, Tor's directory servers use active bandwidth probing to measure and record the bandwidth of each onion router. This generates non-data traffic which causes additional overhead. Tor has to rely on self-advertised bandwidth values if no probing data is available for that specific Onion Router. This may lead to misclassification or may potentially be used by an adversary to classify a router as an entry node [24].

III. Application Based Attacks

Attacks at the application level are generally not a sign of a weakness of the anonymity network, but they occur through

Fig. 4. Alice's IP address is sent through the anonymity network.

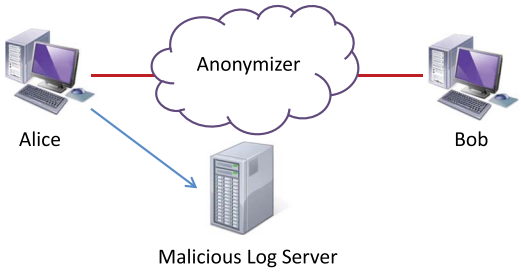

Fig. 5. Alice is tricked to make a connection outside the anonymity network.

the use of an application that does not take privacy into consideration. Application level attacks on anonymity networks attempt to gain the target's real IP address. For instance, with concerns that the Tor network was being used by pedophiles, the Metasploit Project [35] provided a decloaking engine that aided with the de-anonymization of such criminals [36], [37].

The application level de-anonymizing tools typically differ on the avenue of delivery of the target's IP address. They either leak identifying information through the anonymity network or bypass the anoymizer proxy altogether [36], [37]. One of the delivery methods is to send the IP address or identifying information through the anonymity network as in Fig. 4. This is possible because anonymity networks do not filter the actual data being sent by the user [30]. Another approach to leak user identity is through a connection established outside the anonymity network. This involves the application bypassing the local proxy and directly connecting to a server waiting to log the information as in Fig. 5. In this case, a malicious destination, i.e., Bob, or a malicious exit relay, e.g., Mallory, has tricked an application on Alice's computer to establish a direct connection to a logging server after Alice's initial connection through the anonymity network. Alice's real IP address can be logged and her privacy is compromised when the application establishes the direct connection.

In the following, we explain various applications that can be used to leak identifying information from a user of an anonymity network using either approach.

A. Plugins

When browsing the Internet anonymously, various plugins can be used to compromise a user's privacy by bypassing the proxy settings of the client. For instance, the Flash and the QuickTime plugins can be utilized to establish a direct connection to a tracking server. This, of course, would bypass the local proxy settings of the client allowing the attacker to log the victim's real IP address. The main difference between the two plugins is how they achieve this.

Flash contains a standard socket function in its API that establishes a direct connection. The Flash executable ignores any proxy settings in the browser and establishes a separate TCP connection directly from the victim's computer to the specified server.

Similarly, QuickTime allows the setting of a parameter to establish a direct connection to watch a video. The same approach can log the victim's IP address.

We were successful in leaking user identity through a connection established outside the anonymity network using plugins. Using a tracking Apachi server, we obtained user IP when Flash content was executed at user's browser after the initial connection through the anonymity network. Even if the user is connecting to the server via an anonymity network such as Tor or I2P, Flash bypasses the local proxy settings and opened a direct connection to our server.

The Tor Browser Bundle blocks all plugins including Flash and QuickTime. If properly deployed all such active content will be prevented from leaking user information over the anonymity network. However, a victim can be tricked using social engineering to enable plugins when they visit a tracking website and either method can compromise the user's privacy.

B. DNS

The DNS method relies on the SOCKSv4 and SOCKSv5 protocol's inability to properly handle a DNS look up through Tor network that uses TCP protocol [36]. A malicious web page may provide a link that consists of a unique identification number appended to the domain name in use. Because a unique domain name query will be provided to each individual anonymous user, anonymous user's client is triggered to perform a DNS look up. The client application then bypasses the proxy settings and performs the look up through the client system's local DNS server. An authoritative DNS server can log all incoming requests and the client's local DNS server can be identified for further investigation.

This mechanism will not succeed if the Tor or I2P clients are correctly set up to processes DNS queries. This was particularly an issue with Tor, as SOCKSv4 and SOCKSv5 might not handle DNS resolution properly. Hence, Tor provides `Torsocks` tool to resolve DNS queries through the onion proxy. Additionally, pre-fetch feature of browsers can leak information as DNS resolution requests might be sent outside the anonymity network. We were able to utilize the DNS resolution attack against the Tor network when SOCKS version was not SOCKSv4a and `Torsocks` was not enabled.

C. Java Applets

The Java applet approach consists of two tools that use the Java API against the client [36]. The first part of the tool uses a function to resolve host names in the Java API. If the host name in the function call does not match that of the web site providing the applet, a security exception is raised. However, this does not deter the client system from performing the DNS look up. The proxy is bypassed in this process and client identifying information is leaked.

The second method used in the Java applet based attack uses UDP packets. If a UDP packet is sent using a Java applet, the proxy specified by the web browser is bypassed and a UDP packet is sent directly to the destination. A tracking server can be set up to collect UDP packets, which can contain a unique identifier for the anonymous user. Hence, the client system's public IP address can be revealed.

The final approach with Java is rather direct. The Java API allows the applet to look up the host name and IP address of the client similar to executing the `ifconfig` or `ipconfig` commands on the client system. Because the function call is executed on the client's system, it will reveal the network settings of that machine regardless of the proxy settings. The applet may then transmit this information through the Tor socket as regular data.

All of these mechanisms utilize Java applets, but when properly deployed Tor browser bundle blocks Java and prevents such leaks. However, a victim can be tricked to enable Java to view online content on a malicious website and reveal his/her IP address. We were able to obtain user's IP address when the user run Java code that looked up the local IP address and sent it through the anonymity network. Similarly, we were able to get a direct connection to a tracking server after initial connection with our web server that sent a Java code. As indicated, the best mechanism to protect against these vulnerabilities is to block Java code as Tor browser bundle does.

D. Active Documents

A web server can identify the clients connecting to the server even if they are using anonymity networks by transmitting data via active documents to the user's system.

For instance, Acrobat forms allow one to embed JavaScript in a Pdf file. By getting Field object, one can trigger JavaScript using these objects and perform an action. An attacker can define desired functions for these buttons by giving directions with instructions embedded in these fields. For instance, a "recolor" button may, in addition to recoloring an object within the document, direct the user to a specified tracking server.

Many actions can be defined for these fields, such as "Open a file", "Play a sound", "Execute a menu item". The user is unaware of what will actually happen if s/he clicks on the button which might be named anything. Using this approach an attacker can direct a user who downloaded the file through an anonymity network to any tracking server to infer user identity through a new connection that bypasses proxy settings at a later time.

Moreover, newer versions of Adobe Acrobat have a functionality to embed flash, sound and video files. An adversary can leak user identity by sending a pdf file that plays a streaming video on a tracking server. After starting to play the video, the video tracking server continues to communicate with the user bypassing established proxy tunnel. Again a webpage can direct the user to any malicious server that may be able to identify the identity and reveal the location of the user.

Additionally, macros, a series of commands that are recorded so that they can be played or executed automatically, can be used to leak user identity. Many applications, such as Microsoft Word and Excel, support macro languages. When Office is installed on a system, a browser may be set to allow the viewing of documents within it. A query by a document will then bypass the local proxy setting and establish a direct connection to a malicious server. Moreover, this mechanism will also work when the user opens the document at a later time.

In either case, active documents bypass proxy settings and establish a direct connection that reveals user identity. We were able to obtain a user's IP address by sending an active document (i.e., an Adobe pdf file and a Microsoft Word document) through which we established a direct connection for streaming content. Once, a new connection was established to our tracking server with a unique URL, we were able to obtain the user's IP address and match it with previous anonymous browsing. To prevent such attacks, one should be careful with active documents. Actually, Adobe and Microsoft Office warns the user when a file contains active content or has been downloaded from the Internet as they have been abused for various malicious activities.

E. URI Methods

Uniform Resource Identifier (URI), is the generic term for all types of names and addresses referring to objects on the web. Browsers have URI handling, which enables developers to pass application commands through the browser and access software on the user's computer. URIs can be used with different applications, such as establishing a Skype call to a person directly from a website. Overall improper handling of URIs by any application can be exploited to leak user identity.

In addition to direct launch of different applications through a URI, URIs might be used to exploit vulnerabilities in applications handling them. Because many browser implementations do not properly sanitize the data passed via the URI, e.g., with quotes and null bytes, it is possible to circumvent the browsers' ability to handle the incoming data and slip out of the sandbox. Hence, an attacker may be able to access the local file system of the victim.

For instance, iTunes `itms://` can be exploited to leak user identity. If iTunes is installed on the target system, the web server may provide a link to a page containing `itms://`. The protocol handler will then open the iTunes application using the link specified. This will establish a direct connection to the specified domain where the tracking server can log the user's real IP address even if anonymity network is correctly deployed.

Moreover, Skype can be used to leak user identity. Skype has recently released a security bulletin to address a remote vulnerability. This vulnerability was due to an error in the handling of `file://` URIs. By convincing a user to click on a specially crafted `file://` URI, a remote, unauthenticated attacker may be able to execute arbitrary code on the victim's system. Such a code may reveal the user identity even with correctly deployed anonymity network.

Similar to Skype, Apple QuickTime is prone to a vulnerability that allows remote attackers to launch arbitrary applications and access files. The issue arises because of

improper handling of the `file`: URI. The data is passed to `url.dll!FileProtocolHandler` and can result in the processing of non-HTTP file types, if the file type in a URI supplied via the `qt:next` attribute is not recognized by the application. By supplying malicious QuickTime content to be processed by a vulnerable user, an attacker can exploit this vulnerability. The attacker can execute arbitrary malicious codes on the victim's computer and may gain unauthorized remote access control.

As these vulnerabilities were patched, we were not able to utilize these particular exploits. However, we were able to establish a direct call from the user's system via Skype. When Skype is integrated into the browser, a click on a phone number initiates a call. Using clickjacking explained in Section III-H, we were able to get the user make a call to our Skype user. This attack would require collaboration from the Skype to obtain the real identity behind a user's Skype ID. Such an attack can be successful against any VoIp application integrated into the browser as it simply initiates a phone call that can be tracked back.

F. Code Injection

Code injection is the exploration of a computer bug that is caused by processing invalid data. Code injection technique can be used to get the history and cookies of the web browser by injecting HTML or script into the web browser of an anonymous user. An adversary may get identifying information from the user by executing malicious script through a vulnerable website.

Moreover, in anonymity networks, a malicious exit relay can inject HTML tags, scripts or flash as a component of the relayed webpage that can identify the user [38]. In either case, a new connection to a predetermined server is initiated when the user runs the injected component on his/her client and the attacker may obtain the identity of the user. These attacks require knowledge of a vulnerability in user system.

G. BitTorrent

A mechanism to leak user information is through peer-to-peer BitTorrent applications. The BitTorrent application Alice is using can send a packet containing her IP address and compromise her identity as in Fig. 4. It was discovered that some BitTorrent applications reveal a user's identity through the initial registration of peer-to-peer client with the torrent tracker [39]. This occurs despite specifying the use of a SOCKS proxy within the peer-to-peer application. When a user joins a torrent to download a file, the application must announce itself to the tracker specified in the torrent file (unless a distributed hash table is being used). This announcement is done in the form of an `HTTP get` request. This request contains the necessary information to join the torrent and alert the tracker that the client is either looking to share the file or download the file. However, in the torrent specification, the request contains the source IP address as an optional field. Thus, depending on the application, the user could be sending his/her real IP address directly to the torrent tracker.

Fig. 6. Facebook like clickjacking.

The BitTorrent based information leakage has been expanded in [40]. Because Tor uses a single circuit to transmit data over multiple streams, the information gained in previous attacks was used to link specific circuits to an IP address. For example, if the IP address of a user is uncovered during BitTorrent communication, any other traffic through this circuit can be identified. BitTorrent also allows the victim to be tracked across other exit relays controlled by the attacker because of the 20-byte peer identifier used in the communication.

The researchers [39] found multiple bittorent implementations that were leaking the identifying information. However, from our own recent testing, we have found that many of the listed applications have since dropped the optional field from the communication and fixed the vulnerability.

H. Clickjacking

Clickjacking is a malicious technique of tricking a web user to click on a button or a link on a different page from what the user perceives s/he is clicking on. The attacker uses transparent layers to overlay a page on top of a regular page. By this method, the attacker may route a user to another page. An adversary can also direct the request of the user to a malicious server using this method to collect information about the user.

An exploitation of this attack can be done by abusing Facebook's "Like" and Google's "+1" functionality. JavaScript code may seek to make a user "Like" or "+1" a unique link on their Facebook or Google profile via clicking on any link on the malicious website. All the attacker needs to do is to include the JavaScript code to the webpage and create a link on the website. When the user clicks on the link, JavaScript embedded in the webpage automatically makes the user "Like" or "+1" a unique URL as seen in Fig. 6. The assumption with this attack is that the user is already signed into Facebook or Google account, which is often the case for a user with Facebook or Google account.

In either case, malicious website can provide a unique link to each anonymous visitor so that they are identified. However, the attacker would need to get the information from Facebook or Google either legally or crawling their database. We were able to implement both methods and obtain user "Like" or "+1" over the anonymity networks. To achieve the attack, we setup a webpage where a JavaScript code was embedded under a unique link. When the user accessing our website clicked on any link on the webpage, JavaScript embedded in the webpage

automatically made the user "Like" or "+1" a unique URL that we defined. By accessing the profile who accessed this URL, we can get the identity of the user with the help of respective provider.

I. Software Vulnerabilities

Flaws in software libraries that an application or a protocol uses may leak identifying information even with anonymizer technologies.

Recently, a security bug in OpenSSL cryptography library, named Heartbleed, was discovered [41]. OpenSSL is a widely used open source implementation of the SSL and TLS protocols that provide security and privacy of the communication for applications. The HeartBleed bug is a result of a missing bounds check in the handling of TLS heartbeat extension. Arbitrary data in the memory of the systems having vulnerable version of the OpenSSL may be reached by anyone on the Internet. This might allow an attacker to obtain private keys, eavesdrop communications, and steal private data from the services and users even if the user communicates through an anonymized network.

Anonymous communication systems that use OpenSSL have also been effected from this bug [42]. In their blog post, Tor announced that vulnerable clients and relays might be induced to send sensitive information. Using the HeartBleed attack, Tor relays might leak their onion keys and relay identity keys, which allows an attacker who accesses these keys to impersonate the effected relay. The developers of Tor suggest users to update their OpenSSL packages, discard all the files in keys and restart Tor to generate new keys. Because I2P does not use OpenSSL, it was not vulnerable to Heartbleed.

J. Defense Mechanisms Against Application Level Attacks

Attacks at the application level are generally not a sign of weakness of the anonymity network, but they occur through the use of an application that does not take privacy into consideration during implementation. These attacks are possible because of either the carelessness of the user or the application developer.

A common theme among these attacks is the use of active content delivered via web pages. While active content allows for a more dynamic website and user experience, they are detrimental to anonymizer technologies. Even though these issues have been addressed by the *Tor Browser Bundle* by blocking any active content, a victim can be tricked into enabling such plugins or opening active documents.

Most users are not aware of the technical aspects of the anonymizer technologies or active content and may think of blocking active content as nuance. It is essential that users are made aware of the vulnerabilities through active content when using anonymizer technologies.

Moreover, any content that is downloaded can be malicious. For example, a Word or Pdf document downloaded when the *Tor Browser Bundle* is enabled can still establish a direct connection to a tracking server at a later time. The user must ensure s/he is offline when working with such documents, if privacy is a concern.

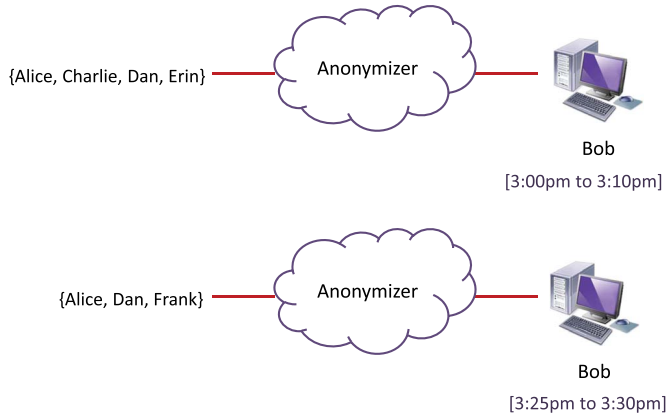

Fig. 7. Intersection attack: The resulting user subset after two observations is {*Alice*, *Dan*}.

IV. Network Level Attacks

In this section, we discuss network level attacks on anonymity networks. Each attack is categorized based on the method used, but many approaches combine different mechanisms to identify the user. Moreover, in some instances, the distinction between the approaches is blurry and we put the mechanism under the closest category. Because of its popular usage among the anonymity networks, most of the surveyed studies are conducted to compromise the anonymity of Tor network. However, these approaches would be successful against I2P unless they are Tor protocol specific.

A. Intersection Attacks

The goal of an intersection attack is to narrow down the possibilities of suspects among a monitored user set. In principle, the attacker may reduce the victim's anonymity group to identify the user behind a particular communication session. For instance, if Mallory wants to figure out who is anonymously communicating with Bob, she can observe Bob's traffic and know during which time periods s/he is being contacted. Knowing the communications are coming from a specific anonymity group, Mallory can look at the users in the anonymity group during Bob's online sessions and identify a subset of suspects. Using this approach, Mallory can further reduce the suspected users through continued observation of the anonymity group during Bob's communications. Fig. 7 present such anonymity group reduction using two communication sessions of Bob. Mallory may eventually reduce this set to a single element of Alice.

Intersection attack is discussed in [43] while enhancing anonymity systems. The analyzed attack is based on profiling a user of the anonymity system. The authors state that users generally follow certain patterns in their browsing. This includes sites visited, online and offline times, email services used, etc. By looking up the set of active users in the system, the attacker can use an intersection of these groups to narrow down an initiator of the communication.

A **predecessor attack** is described in [44]. In this attack, to passively log possible initiators of a stream of communications,

a set of relays work together. The attacker, tracks an identifiable stream of communications over a number of rounds. Any relay that sends a message which is a part of the tracked stream is logged by the attacker at each round. The authors make some assumptions to make the attack successful. For instance, the set of relays utilized by the user to forward his/her message are chosen randomly. Also it is assumed that, the user communicates with the destination in every round. The authors claim that with sufficient path reformations, the attacker can see the initiator, victim, more often than any other user and can decrease the set of initiators to a single user who is victim. An improved and successful **timing intersection attack** was presented against Tarzan and Crowds (which are defunct now) [45]. The authors were able to show that an intersection attack does not necessarily have to be performed by a patient attacker.

This type of attack is also discussed in the Tor design [30]. To address this attack, a user's path is rebuilt approximately every *ten minutes* in the Tor network. Timed circuit use makes an intersection attack difficult. Assuming Mallory is monitoring through a malicious exit relay, she only has about ten minutes to profile Bob's potential peers before a new circuit is built by Alice where Alice would most likely have a new exit relay. Moreover, the *larger the crowd becomes*, the more difficult it is to profile a user. As anonymity networks such as Tor and I2P become popular, its users are essentially hiding among a larger crowd.

A similar **packet counting attack** was presented against Freedom network and PipeNet (which are defunct now) [46]. As anonymity networks such as Mix introduced random delays to prevent timing analysis, this attack focuses on monitoring the packet stream between users. For instance, a malicious attacker Mallory would monitor Alice and Bob's communications to count the incoming and outgoing packets. Mallory might identify communication link between Alice and Bob, if she observes a matching pattern of packet stream between both. Although, the authors focus on Freedom network and PipeNet, the presented attack can be adopted against other anonymity networks.

The packet counting attack is not feasible against the Tor and I2P network as they add random data into the communication session to thwart attacks based on traffic flow analysis. Moreover, as multiple users use the same entry and exit relays in both anonymity networks, matching ingress packet counts to egress counts is not possible by observing traffic flow as an outsider.

Analysis: The *intersection attack* requires a long communication session between the victim(s) and hence cannot work when there is a short session. As Tor and I2P uses a circuit for up to 10 minutes, the attack is very unlikely against Tor and I2P users.

Additionally, as the crowd becomes larger, the likelihood of profiling a victim becomes more difficult. Given that a malicious user monitors k relays among n relays, the attacker has a probability of $(k^2-k)/(n^2-n)$ to observe both the entry (i.e., (k/n)) and exit (i.e., $(k-1)/(n-1)$) relays on a targeted victims communication session. As of this writing, there were over 2,500,000 daily users of Tor and an average of 6,500 relays (excluding 3,500 bridges which make it harder to track an individual). Assuming each user establishes one circuit, then the probability of de-anonymizing "any" user by a malicious user controlling k relays is approximately $(k^2-k)/16.9$ while uncovering a targeted user is approximately $(k^2-k)/4.2x10^7$. Similarly, at the time of this writing, I2P had an estimated 35,000 users while majority were also relays. Hence, the probability of de-anonymizing a targeted user through k malicious relays is approximately $(k^2-k)/1.2x10^9$

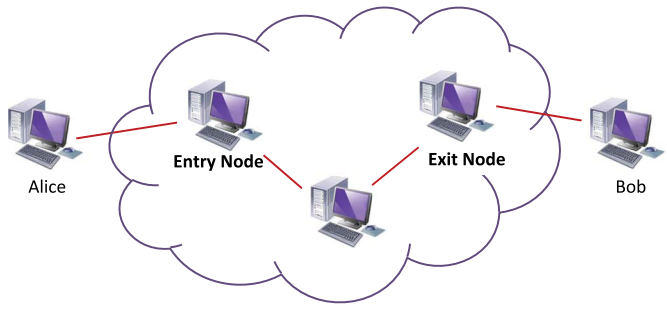

Fig. 8. Multiplication Attack: Attacker controls the entry and exit relays in the circuit.

Note that even though the monitored destination (e.g., Bob) might be known, its communication with the exit relay cannot be used to match entry relay traffic profiles. Hence, the attacker should be able to monitor all relays that can become an exit relay to profile a destination.

Finally, as the attacker relies on monitored traffic, it would not be able to differentiate a victim's traffic if the victim also acts as a relay for other users in the anonymity group. Hence, a Tor or I2P user can protect themselves against intersection attacks by serving other users.

B. Flow Multiplication Attacks

A flow multiplication attack occurs whenever an attacker introduces/modifies packets (or cells in the case of Tor) in the victim's path for the purposes of recognition by a subsequent relay in the flow. This can occur in either direction of the path. One can introduce new cells into the victim's circuit by injecting newly fabricated cells or by sabotaging existing cells to trigger new cells. In either case, the attacker is required to maintain a presence at more than one location of the circuit. The anonymity of a user could be compromised by controlling both the entry relay and the exit relay of that user's selected path as seen in Fig. 8. In this example, a malicious user Mallory controls both entry relay and exit relay of circuit between Alice and Bob and can launch multiplication attacks at either relay and observe the changes at the other relay to identify the communication link between Alice and Bob.

Researchers have accomplished this attack through four methods: duplicating an existing cell, deleting a cell, modifying an existing cell, or inserting a new cell [47]. Because the entry relay knows the source of the TCP stream, e.g., Alice, and the exit relay knows the destination of the stream, e.g., Bob, the two relays can work together to de-anonymize the user behind the connection by triggering actions. These methods must be used while TCP stream data is being sent over the network during the initial circuit building phase. The **replay method** works by

sending a cell that has previously been sent. By duplicating a cell, the AES counter in the circuit is thrown off. This causes a decryption error which propagates to the exit relay. Even though the packet will fail integrity checking, it can be used to uncover the source. The cell **deletion method** works similar to the replay cell. However, in this case the first relay cell is deleted from the stream while all following cells are sent as regular. The error is noticed by the relay and forwarded to the exit relay where identification of the source can occur. The **modification method** can work in a similar manner. The attacker controlling the entry relay can modify the first byte of data in the encrypted payload to zero. This will similarly cause a cell recognition error at the exit relay. The **injection method** cell works much like the modified cell in the sense that the attacker inserts random data into a cell and the error propagates to the exit relay allowing for identification of the source.

The developers of Tor consider the mentioned flow multiplication attack as a **tagging attack** in the original design paper [48]. They also emphasize that this attack carries the risk of being noticed by the targeted user. As of this writing, we can find no evidence that this attack would not work against the current implementation of the Tor network. However, this type of attack has become very difficult to mount against a specified target as it requires the attacker to control both the entry relay and the exit relay on the path of a targeted user. This is very hard to achieve without a significant number of Tor relays being controlled by a malicious party.

Another attack that can be launched with the control of both entry and exit relays is by **encoding attack** where signals are transmitted via cell timing [49]. The attack starts when the exit relay receives return data from the end point of communication, e.g., Bob. The malicious exit relay will then log the IP address and port number of this endpoint and link it to the current circuit. The cells are stored in a queue until the exit relay flushes the data into the network. When a write command is called, the cells are sent to the next relay in the circuit. In the case of a three-hop circuit, the data is sent to the middle relay, which is the only relay in the path that is not controlled by the attacker. By modifying the number of cells flushed to the network, the attacker can encode a signal that can be recognized by the accomplice entry relay. For instance, flushing three cells would indicate an encoded "1" bit, and flushing one cell would encode a "0" bit. Meanwhile, the malicious entry relay will log the stream to compare the encoded signal with the exit relay. Thus, by modifying the exit relay to store cells until the proper encoding is achieved and modifying the entry relay to record the signal, an attacker could identify the user of the marked stream, e.g., Alice.

There is no indication that the *encoding attack* will not work with the current version of Tor as well. However, this attack would be more suitable for a smaller network. As Tor grows with more users offering their systems as relays, the probability of controlling both the entry relay and the exit relay becomes much lower as analyzed at the end of this subsection. Moreover, the attacker needs to identify a very large set of orthogonal short coding patterns so that individual streams can be marked uniquely. With a large number of flows through malicious exit relays, the attacker would need longer coding patterns which would considerably increase the end-to-end delay. This might cause the user to choose other circuits.

A similar **flow multiplication attack** can be launched by injecting HTML code at the exit relay that will trigger a known pattern at the entry relay [50]. Assuming that the attacker has control of both the entry relay and the exit relay on the path between Alice and Bob, attacker can modify replies from Bob to detect the communication link between both. For instance, when Alice requests a web page from Bob's web server, the malicious exit relay injects HTML code with non-visible images. When Alice's web browser receives the HTML page, it requests additional images through the exit relay. Image requests of this nature will generate known patterns that can be observed at the entry relay. As the attacker knows injected pattern at exit relay, it can link Alice to Bob once it observes the pattern at the entry relay.

As of this writing, we cannot find any information indicating that the *flow multiplication attack* would not work as advertised with the current version of Tor. However, as with similar attacks, controlling both the entry relay and the exit relay in the path difficult (as analyzed below).

Finally, a similar **packet size based covert channel** attack on the `anonymizer.com` anonymity network is proposed in [51]. Authors point out that HTTP packets are either 1500 bytes (largest allowed by HTTP) or smaller (typically left over data of a web object transfer). They found that second group (i.e., leftover packets) could be modified to embed a signal to be received on the other end of the anonymity network. To be successful, the attacker needs to control replies sent to the victim and monitor the victim's incoming circuit. For example, a malicious web server would embed a signal into its replies via different packet sizes and attacker would look for that signal at victim's incoming communication. Once the attacker identifies the same packet size pattern at the monitored circuits, it can identify the victim.

In addition to the challenge of controlling destination server and monitoring the entry relay, the *packet size based covert channel* attack only works in an anonymizer network that does not modify the size of the packets. Since current anonymity networks such as Tor and I2P use a uniform packet size, this attack is not possible on them. **Analysis**: Given that a malicious user controls k malicious relays among n relays, the attacker has a probability of $(k^2-k)/(n^2-n)$ chance of controlling both the entry (i.e., (k/n)) and exit (i.e., $(k-1)/(n-1)$) relays on a targeted victims path. If there are f flows then the attacker's chance of de-anonymizing "any" user is $f(k^2-k)/(n^2-n)$. Hence, with current usage of Tor, similar to the Intersection attacks, likelihood of de-anonymizing "any" user by a malicious user controlling k relays is approximately $(k^2-k)/16.9$ while uncovering a targeted user is approximately $(k^2-k)/4.2x10^7$. Similarly, at the time of this writing, I2P had an estimated 35,000 users while majority were also relays. Hence, the probability of de-anonymizing a targeted user through k malicious relays is approximately $(k^2-k)/1.2x10^9$

Hence, while it is easy to uncover a random victims connection, it is very hard to target a specific user with these mechanisms that require control of both entry and exit relays. Furthermore, even if the attacker were to use a resource

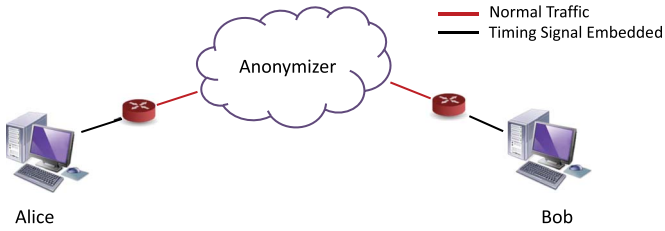

Fig. 9. Timing attack: The timing signal is embedded into the flow.

such as PlanetLab (www.planetlab.org) that currently has about 1,000 nodes, launching a targeted attack would be difficult (approximately 1.77% with 1,000 malicious relays in addition to 6,500 existing relays). Users would also become suspicious when they see a list of relays all hosted by such a source.

An attacker can exaggerate the actual bandwidth of the malicious relays to tempt victims to use the attacker's relays [50]. Deploying this attack would also tempt other users (currently 2.5 million for Tor) in addition to targeted user. Because the relay has suggested it has more bandwidth than it can actually provide, this would overload the relay itself and prevent it from successfully launching an attack.

C. Timing Attacks

As user traffic flows through a particular circuit, the timing pattern of the packets can be used to identify the user. An adversary observing the incoming and outgoing stream of data can link the source and destination and compromise the end points of an anonymity communication.

Although timing analysis attack is discussed in earlier studies, it was first launched successfully against Skype traffic which encrypts the data over a peer-to-peer network [52]. Authors also show that the approach can be applied against www.findnot.com anonymity network (which is defunct now). The purpose of the attack is to determine if user Alice and Bob are communicating using Skype. That is, Mallory suspects that Alice and Bob are communicating, but cannot confirm this because one or both are using an anonymity network to provide unlinkability. Through the inter-packet timing of selected packets, the authors were able to apply a watermark to the packet flow that allows them to link the communication between Alice and Bob without disrupting the real time service provided through Skype as seen in Fig. 9.

Similar timing attacks can be deployed against other anonymous communication networks either from collaborating malicious server or exit relay. However, a significant limitation is that the attacker should know both users of the communication, e.g., Alice and Bob, a priori. The attack may then identify a communication link between the two even if they are using an anonymity network.

The timing attack on Skype has been refined further by [53]. In this case, packet departure times are manipulated by relays designated as Call Markers. Each call marker, deployed at the various edge routers, receives a specific signal function from the entity referred to as the Call Tracker. This essentially provides an ID for that Call Marker. Then, through the use of a Discrete Fourier Transform, the adversary may identify the pattern and know which edge router the traffic originated from. Thus, the end point of the communication under investigation can be discovered. The timing attack, even though not mentioned by the authors, can in theory be launched against the anonymity networks. However, the requirements for de-anonymizing a targeted user is significant as the attack requires cooperation from the ISPs that may host possible suspects to modify delivery time. If there are no suspects for the other end of the communication, then every possible ISP should cooperate with the de-anonymization. Even though every router on the path is not required to participate in the attack, we need a global modification capabilities to de-anonymize the communication link.

A similar **timing attack** with the same principle is presented to identify a communication link between two suspects [54]. As the adversary cannot be certain that Alice and Bob are communicating over an anonymity network, the authors rely on a mark generator and a mark recognizer to determine if Alice and Bob are, in fact, communicating. The mark generator inserts a signal in the communication that can be recognized on the other end of the communication by the mark recognizer. The recognizable timing signal is inserted through the use of a chipping code, referred to as a pseudo noise code. If the recognition system observes this signal within the stream between Alice and Bob, then the communication between them is detected.

With this version of the timing attack mechanism, we have similar issues as the previous timing attacks. The attack will work if the attacker has suspect for the communication, which is not always the case. For example, if Alice is visiting a website that belongs to the attacker using Tor to hide her identity and location, the attacker does not actually know who Alice is and this attack is ineffective. The attacker must have some kind of hunch of where Alice is so the mark recognizers can be properly placed. Overall, this attack is useful in the event that both parties are known and we need to verify the communication between them.

Another study which can be categorized under timing attack examines **traffic flow transformation** methods and how it can be countered in an attack on anonymous communication [55]. The authors divide the transformations as intra-flow transformations and inter-flow transformations. The intra-flow transformations encompass methods that add or remove packets from the network flow in order to better provide anonymity. In the case of Tor, repacketization is used in the form of fragmentation. Each cell is broken down or padded to meet the 512 byte size requirement. The inter-flow transformation occurs when a network flow is either merged with other flows or it is split into multiple flows. Using this knowledge, authors devise a method to divide a timing signal into intervals. By using intervals and redundancy, the signal may survive flow transformations deployed by the network. To effectively pull off this attack, Mallory would insert the signal at the edge of the anonymity network and would decode the signal at the other edge. Authors show that the attack is highly successive with a very low false positive rate on the `Anonymizer.com` network.

However, the attack requires more than 10 minutes of active browsing to be successful. This presents a problem, if we apply this attack to a network that keeps a circuit alive for short

duration. Tor and I2P,in particular, tear down a circuit after 10 minutes and build a new path. Moreover, similar to earlier timing attacks, the attacker requires a suspected source of the communication to perform flow analysis that will capture the flow transformation. The attacker must also have the ability to monitor traffic between suspected user and the anonymity network to detect the embedded signal.

Finally, a timing attack can be launched through applications, i.e., **application based timing attack** [56]. The attack simply triggers the user's web browser into sending a signal over the Tor network that can be detected with traffic analysis. The attack proposed by the authors uses JavaScript to compromise a Tor user's identity with the help of a malicious exit relay and monitoring of entry relays. The malicious exit relay modifies HTTP traffic passing through it, inserting invisible iframe containing JavaScript into requested webpages from a malicious webserver which is controlled by the attacker. While browsing the web through Tor, if by any chance the user selects the malicious exit relay in his/her circuit, the JavaScript would contact the malicious server posting a unique ID. In the meanwhile the attacker performs traffic analysis to compare the signals on each circuit passing through monitored entry relays with the various signals received by the web server. When a Tor client previously connected the malicious exit relay, uses monitored entry relay in a newly formed circuit, the attacker can detect the users communication link by identifying the traffic pattern that JavaScript generates. For most traffic analysis attacks, the attacker must control the exit relay and monitor the entry relay at the same time, but for this attack if a client leaves a browser window open running the JavaScript signal generator, and at any later point chooses a monitored entry relay, then the timing attack can reveal his/her identity.

The same attack can also be implemented with the HTML meta refresh tag. In this version of the attack, the webpage would be modified such that it will automatically be refreshed by the web browser after a period of time. The attacker could generate a desired traffic signal by dynamically varying the refresh delays or the page size each time the webpage is refreshed. The fundamental vulnerability exposed by this paper is not specific to Tor but rather a general problem of anonymous browsing. Most anonymizing systems warn their users to disable active content in their browsers so that malicious codes cannot leak identifying information (see Section III-J).

Analysis: Presented *timing attacks* require a priori knowledge of potential communication sources and destinations. In general, the source is the hardest to identify and this timing attack mechanism would not be effective without suspected user(s). Moreover, random delays and data inserted by anonymity network relays would throw off the attack. Finally, the attack requires timing analysis for a long duration, typically more than the circuit duration of 10 min for Tor and I2P. Hence, applicability of these timing attacks against modern anonymizer systems, i.e., Tor and I2P, is very limited.

D. Fingerprinting Attacks

A fingerprinting attack works much like a detective checking fingerprints from a crime scene. The detective preserves the fingerprints for comparison against a database of previously collected fingerprints of criminals with the hope of finding a match. However, in the case of attack on anonymity, the victim's traffic should be collected for comparison against a previously stored database of traffic signatures. Even encrypted web traffic, while not decipherable without the proper key, can still have a signature to it. For example, if Alice visits www.randomwebsite.com while using the Tor network, it will require similar number of cells for each visit assuming the site has not changed. If Mallory can monitor Alice's traffic to her entry relay, Mallory can record this signature. If Mallory's database of known signatures contains a record for www.randomwebsite.com, she can identify the communication even though she cannot view the actual content of the traffic. Fig. 10 shows Mallory monitoring Alice's traffic and comparing it to her database to determine the destinations. The main difference of fingerprinting attacks from timing attacks (of previous section) is the use of a priori database in the attack.

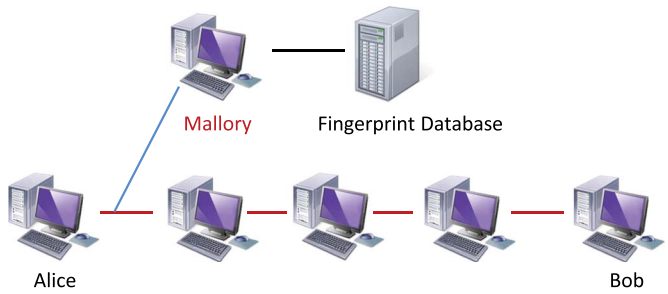

Fig. 10. Fingerprinting Attack: Mallory comparing Alice's traffic to an existing database.

A fingerprinting attack that requires the attacker to control the entry relay of victim's path is proposed in [57]. Authors suggest an attacker may coerce the user into visiting a predetermined site. The attacker will collect packets which build a watermark for a particular site. When the user visits this site, the attacker can match the packet pattern to the previously known pattern.

As of the time of this writing, we can find no information indicating that this attack will not work against the current anonymity networks, i.e., Tor and I2P. Similar to other vulnerabilities, as the anonymity network grows, the success of this type of attack reduces.

A more generic **fingerprinting attack** for use against tunneling methods is presented in [58]. These methods can include OpenSSH, Stunnel, Tor and I2P. There are two phases of the discussed attack. In the training phase, the attacker collects fingerprints for each website the user could possibly visit and stores it in a database. The assumptions taken for this portion are that the attacker knows the anonymity system utilized by the victim and the possible web pages that the victim visits. The attacker should use the same tunneling method as the victim to visit each possible site. From this training phase, the attacker will build a database of fingerprints from the encrypted traffic. The second phase is the testing phase. The assumption here is that the attacker has the ability to observe the traffic from within the tunnel. In other words, the attacker is sniffing traffic

between the victim and the tunnel end-point. The attacker then compares this traffic to the fingerprints stored in the training phase to match the victim's communication with a website in the list.

The discussed *fingerprinting attack* is very effective when it comes to single-hop tunneling methods such as OpenSSH and Stunnel. Authors achieve an accurate identification above 90%. Furthermore, while the single-hop system testing produced a high accuracy, multi-hop systems such as Tor showed this method to be less effective. In particular, identifying Tor traffic of a user had an accuracy below 3% and hence the method is not usable for current onion encryption mechanisms.

Researchers have proposed more effective **fingerprinting attacks against** recent anonymity networks [59]. When the user of the network visited a specific web page, it would require the user to download additional objects within this web page such as images and music. In a standard browser, each object would initiate a separate TCP connection to download that object. These files could then be observed based on file size despite being encrypted. The user could be identified by an attacker observing the communication. However, because current anonymity networks use fixed size messages (e.g., Tor uses 512 byte cells), this attack is not effective by simply observing the traffic created by these objects. The authors offer a modified version of this attack where inter-packet timing is used for fingerprinting instead of the packet size. As each object uses a numbers of cells, the interval (e.g., the inbound traffic for an object without the outbound traffic) patterns are often unique. Authors utilize vectors, i.e., fingerprints, that consists of these intervals for each potential website and the user's first hop. As the user browses the web, the malicious entry relay observes the traffic and builds interval vectors. If portions of this vector matches and indexed website, the attacker can assume the user is connected to the particular webpage.

Researchers have shown successful **fingerprinting attacks** with help of malicious servers [60]. The attack can expose the network identity of an anonymity network user and the relays that are being utilized by the users circuit without the need to control any malicious relay in the network. It is assumed that the attacker controls or creates traffic fluctuation in one end of the communication. Single end controlled available bandwidth tools and a colluding network entity that can modulate the traffic are used to run this attack. As victim communicates with the malicious server, attacker will fluctuate the traffic flow while monitoring available bandwidth of the relays. *LinkWidth*, which is a single end controlled available bandwidth estimation tool that is based on the algorithm behind TCP westwood is used to observe the traffic fluctuation. It provides quick and accurate characterization of the available bandwidth. As attacker knows the traffic pattern, s/he detects when there is a change in the relays s/he is observing and can deduce these relays are being utilized by the victim. In their experiments, authors only identified 11 of 50 circuits, i.e., a success rate of 22%, with one incorrect inference.

A main limitation of this attack is that the attacker controls the target server. Also, the communication should last for a long duration, otherwise, the attacker would not be able to generate detectable traffic pattern with short flows.

Similarly, other researchers have shown effective **fingerprinting attacks** on modern anonymity networks [61]. Authors use a Support Vector Machine (SVM) to perform website fingerprinting based on a variety of features including; total size of all packets in each direction, the size of the document, the number of transmitted bytes, markers for indicating direction changes in packet order, frequency distribution of packet sizes, the percentage of incoming bytes, and total number of packets. Features of two traffic instances induce a distance between them based on how different those features are. Feature similarity can be used as an indicator for whether or not these instances belong to the same site. Both closed world and open world experiments were conducted. In the closed world experiments, the attacker gathered a number of traffic instances from a limited set of sites and the client was only allowed to choose among those sites. Authors correctly identified user traffic in 55% of Tor and 80% of JAP in closed world experiment. Moreover, they achieve a recall of 73% with a false positive rate of 0.05% in the open world scenario with JAP.

Fingerprinting attack against modern anonymity networks is further refined in [62]. The order of incoming and outgoing packets reveals information about the size of objects referenced in a webpage. A web site classifier can be developed based on packet traces from a sequence of page loads performed by the victim. Users can be identified based on the ordering of packets towards a website. Similarity of packet traces is determined using strings where Damerau-Levenshtein distance is used to compare these strings. Hidden Markov Models are also used to model websites where each state corresponds to a page or class of pages on the site. The link structures and probable paths that users will follow among pages when visiting a site are captured by the Hidden Markov Model. An attacker can use these models to determine if a sequence of a victim's page loads are all from the same web site. The experiments showed that the site classifier was able to identify a Tor user visiting one of the 100 web pages with over 80% accuracy.

Finally, an **active fingerprinting attack** that filters data through a malicious entry relay was proposed in [63]. Instead of using TCP/IP packets which are used to transport Tor cells, authors parse the packet data to obtain the underlying Tor cells; i.e., Tor cells are used as unit of data. Authors also remove SENDME cells which provides no extra information to decrease the noise. In their experiments, they achieve a recall rate of 95% with a false positive rate under 0.2%.

Analysis: Given that a malicious user controls k malicious relays among n relays, the attacker has a probability of k/n chance of controlling the entry relay on a targeted victims path. These attacks require the control or observation of traffic from the entry relay of victim to match a fingerprint database. Hence, it is easier to deploy than multiplication attacks that require control or observation of both entry and exit relays.

The assumption that the attacker knows all possible websites the victim could visit is not easy to satisfy. In the web there are tens of billions of web pages (http://www.worldwidewebsize.com). Hence, a very large database of fingerprints would be required to guess the users' destination with any type of certainty.

More importantly, many of websites present dynamic content that would require parallel fingerprinting of such pages. News

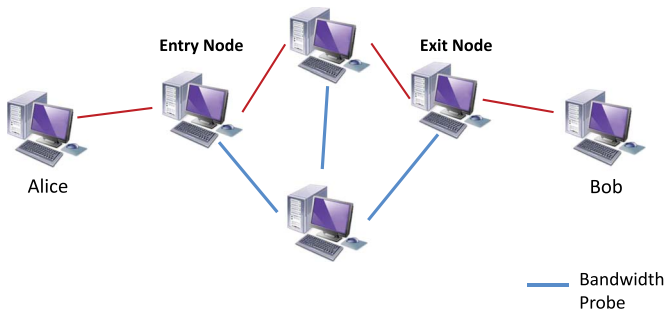

Fig. 11. Circuit clogging attack: Mallory probes relays while fluctuating the traffic.

websites, for instance, provide up-to-minute content that can also be customized for a particular user. Therefore, building recent fingerprints of a large number of websites becomes impractical.

On the other hand, if the attacker was able to somehow guide a victim to a web page uniquely crafted for this purpose, the attacker might be able to de-anonymize the user easier.

E. Congestion Attacks

Circuit clogging, or *congestion attacks*, tend to use a system's or network's resource monitoring capabilities against themselves. While these metrics can be useful in helping the network run at peak performance, they can also be detrimental to the anonymity of the users. The general premise for the attack is to modify traffic in such a way that the victim's path through the anonymity network becomes visible to the attacker as shown in Fig. 11. In this example, a malicious user Mallory may overload certain relays and observe the circuits that are effected by the overloaded relays until she identifies the complete path. Once the path is known, anonymity for the victim is lost.

A **clogging attack** framework against the Freedom anonymity network (now defunct) is presented in [46]. The attacker sniffs the traffic between a certain exit relay and the final destination. While watching this connection, the attacker creates a circuit and clogs selected relays with large number of requests. If the flow between the exit relay and the end point decreases drastically, the attacker can deduce that one of the relays in the circuit belongs to a route that uses the sniffed exit relay. The attacker can apply this method to find all the relays belonging to a certain circuit by changing selected relays. If the circuit that ends at the destination is known, then the entry relay and the user of the circuit might be revealed.

Even though the Freedom network is not in use any more, similar attack might be possible on current anonymity networks. However, as current anonymity networks such as Tor and I2P re-establish circuits every 10 minutes this attack is unlikely to be successful with thousands of relays to probe with congestion.

Researchers have presented the first practical **circuit clogging attack** against the Tor by relying on a malicious server that the victim is connected to [64]. The attack revolves around the observation that a single relay in a path, which is under heavy load, will increase latency for the entire path. In other words, if one of the relays on the users' path is handling a high volume of traffic, the user will notice longer wait times in their communication through the anonymity network. In the presented scenario, the attacker controls a malicious server with which the victim is communicating. The attacker makes a connection to each of the relays to measure the traffic load on the selected relay. The server will generate specific bursty traffic patterns through the Tor network to the victim while monitoring the load on the selected relay. If the overload pattern on the relay matches the generated pattern, the attacker can assume that the monitored relay is on the path to the malicious server. This can be performed on each relay until the entire path of the victim is discovered.

It is important to note that the experiments were performed when the network was in its infancy with only 50 relays. While the resource requirements to implement this attack at the time were reasonable, Tor has since grown significantly to several thousands of relays. Additionally, as the Tor circuits change every 10 minutes, the window of analysis is very limited for the entire network. This is limiting the attack because the bursts of traffic in the paper consists of a few seconds worth of traffic and hence an attacker can only probe 300 relays (assuming 2 sec bursts) before a new circuit is established. The attacker would need to decrease the recognizable burst time to cover every relay (currently 6,500) in the network within 10 minutes. Finally, false positives are very likely when other users increase the load on the network.

An improved variant of the **circuit clogging attack** is presented against Tor in [65]. It is assumed that the attacker controls an exit relay which is used by the victim to access an HTTP server. The attack starts by injecting JavaScript code into an HTML response from the web page the victim is accessing. This script will cause the client to make an HTTP request every second. Each of these requests is handled by the malicious exit relay and a blank response is replied which causes the browser to dismiss the response. Local system time is also included by the JavaScript code to account for the script's potential inability to precisely send requests every second. The adversary then captures the arrival times of the periodic requests performed by the browser. Attacker must also establish a baseline, i.e., measurements without congestion traffic, so a comparison can be made during the congestion attack. In other words, a baseline is taken before and after the attack to ensure that observed changes during the attack are results of the congestion attack, not due to an unrelated change in network characteristics. To launch the congestion attack, the attacker must create a path through each relay under investigation. It is assumed that all Tor relays are suspects and the attacker needs to iterate over all known Tor routers with the aim of finding which one is the entry relay of the victim. For each relay, the attacker constructs a long circuit that repeatedly includes a specific relay in the path. When the circuit is sufficiently long (the authors find 24 hops to be effective, but notes that this length generally depends on the amount of congestion established during the baseline measurements), the attacker uses the circuit to transmit data. If the suspected relay is not a part of the victim's circuit, then there will be no significant change in the measured delays when the attack is running. If the suspected relay is a relay in

the victim's path, then the attacker can identify this based on the delay pattern that s/he observes.

At the time of this attack, there was no restriction for Tor users to establish circuits of arbitrary length (The authors used Tor version 0.2.0.29-rc). Hence, an attacker can specify a variable path length to be established by the client. However, in the current version of Tor, the path length is hard-coded at 3. Additionally, larger number of relays makes it considerably harder to test each potential relay in the anonymizer network.

Analysis: These attacks rely on ability to congest traffic on suspected relays in a user's circuit and observing the effects. In general, detection of the embedded congestion is a challenge as global observation of every relay is needed. Moreover, renewal of circuits every 10 minutes in modern anonymizers such as Tor and I2P limits the success of this attack. As new circuits established, the attacker should keep looking at previously measured relays. Given that these attacks require several seconds to embed and detect congestion, their success against large anonymizer networks (e.g., Tor currently has 6,500 relays while I2P has 35,000 users) is very limited.

F. Resource Attacks

Resource attacks help an attacker direct victims to malicious relays so that other attacks can become more efficient or possible.

The *low-resource routing* attack uses Tor's relay selection algorithms against the network to increase the chances the victim will select the compromised relays [57]. The authors indicate that the attacker can set up the malicious relays and, instead of just hoping that the victim will utilize them when selecting a path, the attacker can exploit the path building process to increase the likelihood the victim selects the malicious relays. There are two relay selection algorithms at work in the Tor network. The first algorithm is responsible for selecting an entry relay for the user's path using two criteria. The first criterion is the relays up-time; i.e., relays that have an up-time greater than the median up-time in the network are preferred. The second criterion is the available resources; i.e., the relays that have more bandwidth to provide will have a better chance of being selected as the entry relay. However, the authors point that the Directory Servers, which keep relay information, do not always verify the resources reported by a relay. A relay simply reports its dedicated bandwidth to the Directory Server while the relay might actually have a lower bandwidth available for Tor traffic. By reporting inflated resources, an attacker can greatly increase the chances of having the victim select the malicious relay. Once the victim has selected the attacker's entry relay and exit relay, many attacks in this study become possible. In a test network of 60 standard relays and 6 malicious relays, the authors were able to control more than 46% of the traffic in the network. In order to ensure the majority of relays are utilized in the network, Tor relay selection algorithm gives less importance to up-time and resources with other relays of a path.

Attacker can also increase the likelihood of being selected by increasing the number of malicious relays either by running several instances over different port numbers or different locations such as PlanetLab like environments that have hundreds or thousands of nodes. These, however, could be detected by careful observations but we are not aware of such an algorithm within Tor or I2P.

Another similar **resource attack** is presented in [66]. The main idea of the attack is to increase the probability of having malicious relays selected in circuit creation by keeping legitimate relays "busy". The attack consists of two phases; loop and compromise. In loop phase, an adversary attempts to keep a significant amount of legitimate routers busy by spinning fraudulent packets. This is achieved by creating a circular circuit which starts and ends at the malicious router and injecting fraudulent packets crafted in a way that will make them spin in the loop. When the attacker completes loop phase, compromise phase is launched in order to reveal anonymous communication. The Mallory injects malicious relays which are not selected as part of any spinning loop. Since most legitimate relays seem to be busy with other circuits, these idle malicious relays will have higher chance to be selected for legitimate circuits. Hence, the anonymity of the communication will be compromised.

Similar to Denial of Service (DoS) attacks (discussed in next section), this attack would require considerable resources of attacker to keep legitimate relays busy. As anonymity networks grows, the contributed resources increase and makes it harder for a malicious user to overload others. However, an attacker with Distributed DoS (DDoS) capability would be more likely to overload the relays so that victims are directed to malicious relays.

As anonymizer software generates a path for a user, the algorithm used to generate the path can be profiled. As certain relays carry more traffic, they can be profiled and targeted [67]. Relays with high availability and bandwidth (called as "super nodes" by authors) essentially form the core of the Tor network. Even though the core consists of 21% of the relays, they carry approximately 66% of the traffic in their measurement study. These relays are important because the path selection algorithm is biased towards these relays. Named as *loop attack* by the authors, the attack begins by identifying the super node subset. With this knowledge, the attacker begins to take relays from this subset off-line (ISP level blocking or DoS). After continuous downtime in the network, users begin to lose confidence in the network. After each cycle, the number of users of the network drops due to the inability to access the network. This, in turn, reduces the remaining users' anonymity set. One of the cornerstones of anonymity networks is that users are "hiding among the crowd". If there is a smaller crowd to hide among, the anonymity level is reduced and the attacker can more easily compromise identities through other attacks.

Analysis: Resource attacks in principle help an attacker direct victims to malicious relays so that other attacks can be launched. Most of the discussed attacks in this section relied on the control of a relay and resource attacks can help attacker direct victims to malicious relays. Even though they depend on other attacks to be successful, they should be monitored so that users with malicious intent are detected. In general, significant activity by a user or a set of related users is easy to detect. However, as far as we are aware, Tor and I2P do not have such mechanisms to automatically alert users.

Anonymizer networks can protect against these type of attacks by measuring observed bandwidth in selection of entry relays. A relay can be selected as an entry node in the circuit only after it proves to be reliable and stable. In Tor, a relay goes through two phases (which takes 8 days) until it can be named 'reliable' to prevent resource exaggeration and 'stable' to enhance communication success [31]. However, a malicious relay can be used in the circuits as an entry relay after eight days with the current implementation.

G. Denial of Service (DoS)

A denial of Service (DoS) attack is not specific to anonymity networks, but it can have a significant impact on any targeted network. In general, a DoS attack on anonymity network can lead users of the network to seek out less secure (i.e., in terms of anonymity) means of communication [68]. In other words, if the user is unable to access the anonymity network, they might use a weaker method to hide their location or no method at all. This, of course, compromises their identity and location.

DoS attacks on anonymity networks can be divided into two categories. The first is a **blanket blocking** that blocks access to the entire network. While no individual relay is actually attacked, access to any relay associated with the anonymity network is blocked. An example of this would be a government's block on access to the relays listed in Tor's directory servers [69]. Because the list of relays in the anonymity network need to be publicly available, a country, an ISP, or a network administrator can easily obtain this list and deny access to the directory servers and listed relay IP addresses to block access to the whole anonymity network.

The counter-measure put forth by the developers of Tor is the use of bridges [70]. Bridges, hosted by volunteers, provide an alternative method of connecting to the Tor network. For instance, a user who is not blocked from accessing the anonymity network can volunteer to have other users connect through their system for access Tor. The bridge's IP address is not listed in the directory server but distributed through private channels. For example, bridge IP address can be sent through specific e-mail addresses or private communications. However, to counter this, several recent mechanisms have been deployed [71]. Certain countries imitate legitimate users and obtain bridge addresses to block them [72]. Similarly, adversaries might sift through e-mails or communications for IP addresses and try to establish Tor circuits to identify bridges [73].

The second type of DoS attack on anonymity networks is a **targeted attack**. This involves the adversary targeting specific relays in the network and taking them offline. Vulnerabilities in software are often discovered and patched yet they are known to exist, even temporarily. In addition to a newly discovered vulnerability, an adversary can launch a Distributed DoS (DDoS) attack from a botnet to block certain relays.

A DoS attack could be launched to bring down super nodes in the network and significantly reduce the performance of the super nodes. According to one estimate, 21% of the nodes in the network handle 66% of the traffic [67]. By targeting these relays, an attacker could have an adverse effect on the quality of overall communication. The authors indicate this attack could reduce the number of users and hence the anonymity of the remaining users can be compromised easier. However, such an attack on super nodes would be highly visible. Even the attack presented by [67] did not bring down the entire core to affect performance. However, an ISP providing service to a significant number of super nodes (2% of regular relays and 10% of super nodes, in the paper) could have a great impact on the performance of the anonymity network. Of course, this attack would be highly visible and likely to be acted upon.

Another **DoS attack against** Tor is to exploit the circuit establishment mechanism in [74]. Whenever a Tor client extends a circuit, it generates one more layer using public onion key of the target router. The target router processes it using its private onion key. The authors indicate that this operational model makes the processing for routers more expensive than generating them. This imbalance can be exploited by malicious client to consume computational resources of a router by continuously generating CREATE cells. A router receiving CREATE cells at a high rate will start discarding packets even coming from legitimate clients. Hence, this will decrease the likelihood of this relay being selected for a circuit while increasing the probability of malicious relays to be selected in the circuit.

Similarly, **Sniper Attack** is a DoS attack that can target Tor relays [75]. The attack utilizes Tor's application level congestion and flow control mechanisms to cause a relay to buffer an arbitrary amount of data in its application queues. To achieve this, a malicious client builds a circuit using a targeted Tor relay as an entry relay and downloads a large file. The malicious client then sends SENDME cells to the exit while ignoring transmitted data. Large number of SENDME cells will signal the exit relay to increase its congestion window while pulling data from the destination. Eventually the target relay queuing the data will exhaust its memory allocation and will be terminated by its operating system. Moreover, an attacker can increase attack effectiveness by establishing a large number of such circuits so that targeted entry relay is exhausted before selected exit relay(s). Authors also discuss possible counter measures against this attack. One of the defenses is, for each relay to enforce a maximum queue size to limit the amount of the memory each circuit can consume. If the queue length becomes greater than the allowed size, the relay might assume a protocol violation has occurred and can terminate the circuit. This would prevent enormous growth of Tor's application queue by a malicious user. Even though authors propose a defense mechanism against parallel sniper attack by randomly terminating circuits when a memory threshold is exceeded. However, the presented mechanism still leads to DoS. It should also be noted that, since security requirements has been implemented to defend against this attack, as of version 0.2.4.18-rc and later, Tor network is no longer vulnerable to regular sniper attack. However, parallel sniper attack is still possible. Note that it is harder to prevent DDoS type attacks compared to DoS attacks.

Finally, researchers have also discussed a **DoS attack** on I2P network by controlling portion of the I2P database [76]. By controlling majority of I2P database, i.e., netDB, an adversary can log database actions for the managed keyspace and can compromise the anonymity of users. The set of relays can be manipulated by exploiting the normal churn in the set of

participating relays or by carrying out a DoS attack to speed up the takeover. Additionally, *Sybil attack*, in which adversary generates a large number of pseudonyms to gain a disproportionately large influence in the system, can be used to take control of the netDB. By leveraging control of the network database, the adversary can then launch an *Eclipse attack*, in which colluding relays are promoted to become part of communication tunnels. This results in services being unavailable or peers getting disconnected from the network. Finally, it is possible for the attacker to link any user with services s/he uses. The attack exploits the protocol used by peers to verify the successful storage of their peer information in the netDB. The storage and verification steps are done through two independent connections that can be linked based on timing. Using the information gathered by linking these two interactions, an attacker can determine which tunnel endpoints belong to specific participants in the I2P network, and deanonymize the users.

Analysis: Denial of Service attacks are a general concern for communication systems and traditional crypto-systems are not able to handle them. For anonymizer networks, the main concern has been to make the service available to users that need it but are prevented from accessing public servers or relays. General focus has been on enabling the service through bridges for users whose connections are monitored and/or blocked. However, in some cases these bridges might also be detected and become inaccessible. In utmost case, a user can depend on a personal bridge that is never advertised to anyone else so that it is not detected.

V. Discussion

In this paper, we have classified attacks on anonymity networks into two major categories; namely, *application level attacks* and *network level attacks*.

A. Application Level Attacks

Application level attacks are often possible because of either the carelessness of the user or the application developer. First requires the user to not reveal any identifying information while communicating anonymously. There is not much the technology can do to prevent such attacks as user should be diligent in their transmissions when they want to be anonymous. The latter is an issue with the implementation of user applications that might leak identifying information. Following is a list of such compromises.

- Bypass local proxy settings: e.g., Java, Flash, and QuickTime
- Protocols exposing user identity: e.g., DNS, iTunes, and BitTorrent
- Active documents: e.g., Word, PDF, and Excel
- Clickjacking: Facebook, Google+, and Skype
- Social engineering: Enabling active content

Lack of proxy support is an issue with anonymizer networks. For instance, Flash and Java ignores the system proxy settings and any connections established are performed directly from the user's location, i.e., bypassing any anonymized proxy. Similarly, QuickTime allows the setting of a parameter to establish a direct connection to watch a video from a web server, which might allow to log the real IP of the user by the server.

Moreover, some applications have *poor proxy support*. For instance, anonymizer developers might have forgotten to perform DNS lookups through the proxy, as it was the case with earlier Tor versions. While the user's application data is properly being sent through the proxy, the DNS traffic is being sent directly from the user to the lookup server. Additionally, an application might transmit identifying information. For example, certain implementations of BitTorent applications were discovered to transmit user IP address over anonymizer network when registering in the system.

Another way to find a user's identity is *active documents* through which injected codes may establish connections to a malicious website that look for such connections. As the user might not have anonymizer proxy all the time, the direct connection to a malicious web server would reveal the user's identity.

Moreover, a user might be tricked into clicking on a unique object that would reveal his/her identity if the user was logged onto certain services such as Facebook and Google+. Such *clickjacking* can even establish Skype connection if the user's browser is configured to call numbers from Skype.

By *social engineering* a user can also be tricked to enable active content which are typically disabled by default with anonymizer tools. Of course such social engineering might leak identifying information from the user as well.

B. Network Level Attacks

Network level attacks on anonymous communication are often out of the control of the end user. The attack generally exist because of either a limitation of the network or a trade off that was made in the development stage. For instance, timing attacks invoke different tradeoffs. A simple method to counter timing attacks is insertion of random traffic payload and random delaying of message transmissions. The downside of this approach is the overhead, not only at the end system, but at the network as well. Additional random traffic and incurred delay degrade the overall quality of service of the anonymity network. Hence, Tor network does not employ these mechanisms to provide low-latency communication. Likewise, timing attacks can be circumvented by establishing a new circuit very frequently. However, that will cause a considerable control traffic and hence Tor and I2P developers decided to keep a circuit for up to 10 minutes, which is well below average time required for the discussed attacks.

Moreover, certain limitations of an anonymity network provide vulnerabilities. For instance, flow multiplication attacks or congestion attacks become less effective as an anonymity network grows. In the earlier stages of any anonymity network, these attacks would be feasible to launch. Most of the surveyed network level attacks were actually analyzed *when the attacked anonymity networks were in infancy stages* with tens to hundreds of relays. With a smaller network, it is easy for an attacker to probe each relay to determine which ones are

TABLE I
NETWORK LEVEL ATTACK COMPARISON

	Study	Control of Any Relay in Circuit	Control of Entry Relay	Control of exit Relay	Control of Destination	Monitor All Relays	Monitor Entry Relay	Monitor Exit Relay	Source Suspect Required	Passive Attack	Improvable with Resource Attacks	Effectiveness: Tor	Effectiveness: I2P	Remedies (for all): Short Timed Circuits	Larger Crowd	Add Random Data/Delay	Fixed Sized Cells	Serving as Client and Relay
Intersection	Berthold et al.[43]								✓	✓		✓	✓					
	Wright et al.[45]		✓						✓	✓		✓	✓	✓	✓	✓	✓	✓
	Back et al.[46]						✓		✓									
Flow Multip	Fu and Ling[47]		✓	✓							✓	✓	✓					
	Ling et al.[49]		✓	✓					✓		✓	✓	✓					
	Wang et al.[50]		✓	✓							✓	✓	✓		✓		✓	
	Ling et al.[51]				✓				✓									
Timing	Wang et al.[52]								✓			✓	✓					
	Yu et al.[54]								✓			✓	✓	✓	✓	✓		
	Wang et al.[55]								✓			✓	✓					
	Abbott et al.[56]		✓	✓							✓	✓	✓					
Fingerprinting	Bauer et al.[57]		✓						✓	✓		✓	✓					
	Herrmann et al.[58]						✓		✓	✓		✓	✓					
	Shi et al.[59]		✓						✓			✓	✓					
	Chakravarty et al.[60]				✓	✓						✓	✓	✓	✓	✓		
	Panchenko et al.[61]								✓	✓		✓	✓					
	Cai et al.[62]								✓	✓		✓	✓					
	Wang and Goldberg[63]								✓			✓						
Congestion	Back et al.[46]	✓							✓			✓	✓					
	Murdoch and Danezis[64]				✓	✓			✓		✓	✓	✓	✓	✓			
	Evans et al.[65]	✓		✓		✓			✓		✓	✓	✓					

being employed by a victim. As the network grows, however, such attacks become infeasible as an attacker cannot complete probing before a new circuit is established.

Table I provides a summary of analyzed attack mechanisms. It provides the underlying mechanisms deployed by each approach, their overall effectiveness against Tor and I2P, and remedies to prevent such attacks. As can be seen in the table, we cannot generalize each feature to a specific type of attack. Although presented in the same category, one attack can require the attacker to be passively monitoring the traffic while the other requires active modification or insertion of patterns into the traffic that is being observed.

Overall, the goal of the *intersection attacks* is simply to narrow down the possibilities of suspects. This attack requires a continuous communication from the victim and hence cannot work when there is a single or few sessions. Moreover, the attack can be practical on a network with a single ingress point at the destination. That is, the attack would not work on services deployed over I2P network where ingress points of web sites change. In addition, the attack requires a long communication between victims, and hence cannot work when there is a short session. Tor and I2P uses a circuit for up to 10 min so the attack is very unlikely against these networks.

In the *flow multiplication attacks*, the attacker is required to maintain a presence at more than one location of the circuit in an anonymity network. Most of these attacks require the attacker to have a control on both the entry and the exit relays. Compared to other attacks, flow multiplication attacks can be improved more with the help of resource attacks. *Resource attacks* help an attacker to lead victims to malicious relays. Even though an attacker does not have significant number of malicious relays in the network, through resource attacks, s/he can keep the legitimate relays busy and increase the chance of her/his relays being selected by victim while building communication paths.

Timing pattern of the packets are used to deploy *timing attacks*. In general, these attacks require the user to insert a marker in the communication link to deanonymize users, which makes these attacks an active attack similar to flow multiplication attacks. It also requires a priori knowledge of potential communication sources and destinations, by this way an attacker could identify the inserted marker in the communication link s/he is tracing. This nature of the attack eliminates the need to control entry/exit relays in the anonymity network.

Fingerprinting attacks mostly require the control or observation of traffic from the entry relay of victim's path to match a fingerprint database. This makes them easier to deploy than multiplication and intersection attacks that require control/observation from both entry and exit relays. However, if we consider the huge number of existing websites, it will be

challenging for the attacker to match the fingerprints in a short amount of time. Moreover, these attacks are applicable when there is a targeted suspected victim.

The general premise for *congestion attacks* is to modify traffic in such a way that the victim's path through the anonymity network becomes visible to the attacker. Hence, it is observed that most of these attacks are active which involves modification or insertion of any element to the communication while monitoring it. Instead of active deployment of malicious relays, the attacker monitors relays in the network to detect embedded congestion. The attack is difficult as the attacker needs to observe potentially all of the relays to detect the one being used in the communication path.

Most of the presented attacks targeted the Tor network. However, in general, they can be applied against I2P as well. Short timed circuits built in Tor and I2P is one of the efficient countermeasures taken by the developers to prevent possible network level attacks. Some of these attacks, such as flow multiplication, timing and congestion attacks, need either active control or monitoring of the relays in the network. Similarly, some of the attacks under intersection and fingerprinting also require the attacker to observe network relays. As the network size gets bigger, it becomes harder for the attacker to compromise these networks as it becomes impossible to profile a user. Random delay or data injection by Tor and I2P is another mechanism to thwart some of these attacks, especially traffic flow analysis. Similarly, as Tor and I2P transmits the data in fixed size chunks, packet size based analysis of messages is not possible. Especially when there is a targeted user that the attacker wants to profile, being a relay while using the anonymity network prevents the user from being traced. Finally, one can utilize multiple paths in transmitting their content for more efficient communication and better protection from watchful adversaries [77].

VI. Conclusion

Privacy of communication has been an essential requirement for online communication. The expanding reliance of Internet communications for our daily lives resulted in growing interest in methods for anonymous communication. Several system designs have been developed with the aim of preserving communication privacy within the shared network environment. On the other hand, growing interest in anonymity systems also triggered the focus on de-anonymization techniques.

In this paper, we first overview the currently deployed anonymity technologies including Tor and I2P. Then, we surveyed the de-anonymization approaches so that we can better understand vulnerabilities in the anonymity networks. The effectiveness of attack approaches rely on certain mechanisms and hence their success vary depending on the attacked anonymity network. Moreover, while some attacks are possible in theory, their effective deployment is prohibitive. Likewise, protection mechanisms for simpler attacks, such as application level attacks are usually provided if anonymizer tools are correctly deployed.

The rapid development of anonymity networks is a sign that they will have much wider applications in the future. A deeper analysis of these systems will help the users seeking online privacy protection. Moreover, as new anonymity mechanisms are developed and new vulnerabilities in anonymity networks are identified, both anonymization and de-anonymization approaches will grow, respectively.

As a future direction, we need to consider potential security vulnerabilities in the anonymizer technologies and online communication applications. As the community is developing Tor and I2P anonymizer technologies, they tend to identify potential security issues that can compromise user privacy. Additionally, vulnerabilities in various user applications need to be monitored to detect potential issues. A bigger issue is the education of anonymizer technology users as they might become a victim to social engineering and enable active content. Even though the current anonymity network are secure, the user's action can easily leak identifying information and make the anonymity technology useless.

References

[1] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–90, Feb. 1981.

[2] G. Danezis, R. Dingledine, D. Hopwood, and N. Mathewson, "Mixminion: Design of a type iii anonymous remailer protocol," in *Proc. IEEE Symp. Security Privacy*, 2003, pp. 2–15.

[3] K. Sako and J. Kilian, "Receipt-free mix-type voting scheme: A practical solution to the implementation of a voting booth," in *Proc. 14th Annu. Int. Conf. Theory Appl. Cryptogr. Tech. EUROCRYPT*, 1995, pp. 393–403.

[4] M. Jakobsson, A. Juels, and R. L. Rivest, "Making mix nets robust for electronic voting by randomized partial checking," in *Proc. 11th USENIX Security Symp.*, 2002, pp. 339–353.

[5] D. Boneh and P. Golle, "Almost entirely correct mixing with applications to voting," in *Proc. 9th ACM Conf. CCS*, 2002, pp. 68–77.

[6] M. Waldman and D. Mazires, "Tangler: A censorship-resistant publishing system based on document entanglements," in *Proc. 8th ACM Conf. Comput. Commun. Security*, 2001, pp. 126–135.

[7] M. Waldman, A. D. Rubin, and L. F. Cranor, "Publius: A robust, tamper-evident, censorship-resistant, web publishing system," in *Proc. 9th USENIX Security Symp.*, 2000, pp. 59–72.

[8] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A distributed anonymous information storage and retrieval system," in *Proc. Int. Workshop Des. Privacy Enhancing Technol.—Design Issues Anonymity Unobserv.*, 2001, pp. 46–66.

[9] R. Dingledine, M. J. Freedman, and D. Molnar, "The free haven project: Distributed anonymous storage service," in *Proc. Workshop Des. Issues Anonymity Unobserv.*, 2000, pp. 67–95.

[10] R. Anderson, "The Eternity Service," 1996, pp. 242–252.

[11] D. Kelly, "A taxonomy for and analysis of anonymous communications networks," Air Force Inst. Technol., Wright-Patterson AFB, OH, USA, Tech. Rep., March 2009.

[12] N. Mathewson and R. Dingledine, "Practical traffic analysis: Extending and resisting statistical disclosure," in *Proc. PET Workshop*, May 2004, vol. 3424, LNCS, pp. 17–34.

[13] C. Troncoso and G. Danezis, "The Bayesian traffic analysis of mix networks," in *Proc. 16th ACM Conf. CCS*, 2009, pp. 369–379.

[14] J.-F. Raymond, "Traffic analysis: Protocols, attacks, design issues, and open problems," in *Proc. Workshop Design Issues Anonymity Unobservability*, H. Federrath, Ed., Jul. 2000, vol. 2009, LNCS, pp. 10–29.

[15] G. Danezis and C. Diaz, "A survey of anonymous communication channels," Microsoft Res., Cambridge, U.K., Tech. Rep. MSR-TR-2008-35, 2008.

[16] Free Haven Selected Papers in Anonymity. [Online]. Available: http://www.freehaven.net/anonbib/date.html

[17] Tor Project: Anonymity Online. [Online]. Available: https://www.torproject.org

[18] I2P Anonymous Network. [Online]. Available: www.i2p2.de

[19] Iran Partially Blocks Encrypted Network Traffic. [Online]. Available: https://blog.torproject.org/blog/iran-partially-blocks-encrypted-network-traffic

[20] M. Ligh, S. Adair, B. Hartstein, and M. Richard, *Malware Analyst's Cookbook and DVD: Tools and Techniques for Fighting Malicious Code*, 1st ed. Hoboken, NJ, USA: Wiley, Nov. 2010.

[21] K. Poulsen, FBI's Secret Spyware Tracks Down Teen Who Made Bomb Threats, Jul. 2007. [Online]. Available: http://www.obscura.com/loki/remailer/remailer.essay.html

[22] M. Edman and B. Yener, "On anonymity in an electronic society: A survey of anonymous communication systems," *ACM Comput. Surv.*, vol. 42, no. 1, pp. 1–35, Dec. 2009.

[23] J. Ren and J. Wu, "Survey on anonymous communications in computer networks," *Comput. Commun.*, vol. 33, no. 4, pp. 420–431, Mar. 2010.

[24] B. Conrad and F. Shirazi, "A survey on Tor and I2P," in *Proc. 9th ICIMP*, Jul. 2014, p. 22.

[25] B. Li, E. Erdin, M. H. Gunes, G. Bebis, and T. Shipley, "An analysis of anonymity technology usage," in *Proc. 3rd Int. Conf. TMA*, 2011, pp. 108–121.

[26] B. Li, E. Erdin, M. H. Gunes, G. Bebis, and T. Shipley, "An overview of anonymity technology usage," *Comput. Commun.*, vol. 36, no. 12, pp. 1269–1283, Jul. 2013.

[27] A. Pfitzmann, T. Dresden, and M. Hansen, "Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management—A Consolidated Proposal for Terminology," 2008.

[28] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing," *Commun. ACM*, vol. 42, no. 2, pp. 39–41, Feb. 1999.

[29] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Anonymous connections and onion routing," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 482–494, May 1998.

[30] R. Dingledine, N. Mathewson, and P. Syverson, "TOR: The second-generation onion router," in *Proc. 13th Conf. UNESIX SSYM*, 2004, pp. 21–21.

[31] The Lifecycle of a New Relay. [Online]. Available: https://blog.torproject.org/blog/lifecycle-of-a-new-relay

[32] A. Panchenko, F. Lanze, and T. Engel, "Improving performance and anonymity in the Tor network," in *Proc. IEEE 31st Perform. Comput. Commun. Conf.*, 2012, pp. 1–10,

[33] J. P. Timpanaro, I. Chrisment, and O. Festor, "Monitoring the I2P Network," Preprint, Oct. 2011. [Online]. Available: http://hal.inria.fr/inria-00632259

[34] zzz (Pseudonym) and L. Schimmer, "Peer profiling and selection in the I2P anonymous network," in *Proc. PET-CON*, Mar. 2009, pp. 59–70.

[35] Metasploit. [Online]. Available: www.metasploit.com

[36] Metasploit Decloaking Engine. [Online]. Available: http://decloak.net

[37] Hacker Builds Tracking System to Nab Tor Pedophiles, Feb. 2011. [Online]. Available: http://www.zdnet.com/blog/security/hacker-builds-tracking-system-to-nab-tor-pedophiles/114

[38] A. Christensen, "Practical onion hacking: Finding the real address of Tor clients," FortConsult, København, Denmark, Tech. Rep., Oct. 2006.

[39] P. Manils *et al.*, "Compromising Tor Anonymity Exploiting p2p Information Leakage," vol. abs/1004.1461, 2010.

[40] S. Le Blond *et al.*, "One bad apple spoils the bunch: Exploiting P2P applications to trace and profile Tor users," in *Proc. 4th USENIX Conf. LEET*, 2011, pp. 2–2.

[41] The Heartbleed Bug. [Online]. Available: http://heartbleed.com/

[42] Openssl Bug cve-2014-0160. [Online]. Available: https://blog.torproject.org/blog/openssl-bug-cve-2014-0160

[43] O. Berthold, H. Federrath, and M. Köhntopp, "Project 'Anonymity and unobservability in the Internet'," in *Proc. 10th Conf. CFP*, 2000, pp. 57–65.

[44] M. Wright, M. Adler, B. N. Levine, and C. Shields, "The predecessor attack: An analysis of a threat to anonymous communications systems," *ACM Trans. Inf. Syst. Security*, vol. 4, no. 7, pp. 489–522, Nov. 2002.

[45] M. Wright, M. Adler, B. N. Levine, and C. Shields, "Defending anonymous communication against passive logging attacks," in *Proc. IEEE Symp. Security Privacy*, May 2000, pp. 28–41.

[46] A. Back, U. Möller, and A. Stiglic, "Traffic analysis attacks and trade-offs in anonymity providing systems," in *Proc. IH Workshop*, I. S. Moskowitz, Ed., Apr. 2001, vol. 2137, LNCS, pp. 245–257.

[47] X. Fu and Z. Ling, "One cell is enough to break Tor's anonymity," Black Hat DC, 2009.

[48] Arma, One Cell is Enough to Break Tor's Anonymity, Feb. 2009. [Online]. Available: https://blog.torproject.org/blog/one-cell-enough

[49] Z. Ling *et al.*, "A new cell counter based attack against Tor," in *Proc. 16th ACM Conf. CCS*, 2009, pp. 578–589.

[50] X. Wang, J. Luo, M. Yang, and Z. Ling, "A novel flow multiplication attack against Tor," in *Proc. 13th Int. Conf. CSCWD*, 2009, pp. 686–691.

[51] Z. Ling, X. Fu, W. Jia, W. Yu, and D. Xuan, "A novel packet size based covert channel attack against anonymizer," in *Proc. 30th IEEE INFOCOM*, Shanghai, China, Apr. 10–15, 2011, pp. 186–190.

[52] X. Wang, S. Chen, and S. Jajodia, "Tracking anonymous peer-to-peer VoIP calls on the Internet," in *Proc. 12th ACM Conf. CCS*, 2005, pp. 81–91.

[53] H. Sengar, Z. Ren, H. Wang, D. Wijesekera, and S. Jajodia, "Tracking skype VoIP calls over the Internet," in *Proc. IEEE INFOCOM*, Mar. 14–19, 2010, pp. 1–5.

[54] W. Yu, X. Fu, S. Graham, D. Xuan, and W. Zhao, "DSSS-based flow marking technique for invisible traceback," in *Proc. IEEE Symp. SP*, 2007, pp. 18–32.

[55] X. Wang, S. Chen, and S. Jajodia, "Network flow watermarking attack on low-latency anonymous communication systems," in *Proc. IEEE Symp. Security Privacy*, 2007, pp. 116–130.

[56] T. G. Abbott, K. J. Lai, M. R. Lieberman, and E. C. Price, "Browser-based attacks on Tor," in *Proc. 7th Int. Conf. PET*, 2007, pp. 184–199.

[57] K. Bauer, D. McCoy, D. Grunwald, T. Kohno, and D. Sicker, "Low-resource routing attacks against Tor," in *Proc. WPES*, Washington, DC, USA, Oct. 2007, pp. 11–20.

[58] D. Herrmann, R. Wendolsky, and H. Federrath, "Website fingerprinting: Attacking popular privacy enhancing technologies with the multinomial naive-Bayes classifier," in *Proc. ACM CCSW*, 2009, pp. 31–42.

[59] Y. Shi and K. Matsuura, "Fingerprinting attack on the Tor anonymity system," in *Information and Communications Security*. Berlin, Germany: Springer-Verlag, 2009, pp. 425–438.

[60] S. Chakravarty, A. Stavrou, and A. D. Keromytis, "Traffic analysis against low-latency anonymity networks using available bandwidth estimation," in *Proc. 15th ESORICS*, 2010, pp. 249–267.

[61] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel, "Website fingerprinting in onion routing based anonymization networks," in *Proc. WPES*, Y. Chen and J. Vaidya, Eds., 2011, pp. 103–114.

[62] X. Cai, X. C. Zhang, B. Joshi, and R. Johnson, "Touching from a distance: Website fingerprinting attacks and defenses," in *Proc. ACM Conf. Comput. Commun. Security*, T. Yu, G. Danezis, and V. D. Gligor, Eds., 2012, pp. 605–616.

[63] T. Wang and I. Goldberg, "Improved website fingerprinting on Tor," in *Proc. 12th ACM WPES*, 2013, pp. 201–212.

[64] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of Tor," in *Proc. IEEE Symp. Security Privacy*, May 2005, pp. 183–185.

[65] N. Evans, R. Dingledine, and C. Grothoff, "A practical congestion attack on Tor using long paths," in *Proc. 18th USENIX Security Symp.*, Aug. 2009, pp. 33–50.

[66] V. Pappas, E. Athanasopoulos, S. Ioannidis, and E. P. Markatos, "Compromising anonymity using packet spinning," in *Proc. 11th ISC*, Sep. 2008.

[67] C. Li, Y. Xue, Y. Dong, and D. Wang, "'super nodes' in Tor: Existence and security implication," in *Proc. 27th ACSAC*, 2011, pp. 217–226.

[68] N. Borisov, G. Danezis, P. Mittal, and P. Tabriz, "Denial of service or denial of security? How attacks on reliability can compromise anonymity," in *Proc. CCS*, Oct. 2007, pp. 92–102.

[69] Tor Partially Blocked in China. [Online]. Available: https://blog.torproject.org/blog/tor-partially-blocked-china

[70] Tor Bridges. [Online]. Available: https://www.torproject.org/docs/bridges

[71] Research Problems: Ten Ways to Discover Tor Bridges. [Online]. Available: https://blog.torproject.org/blog/research-problems-ten-ways-discover-tor-bridges

[72] P. Winter and S. Lindskog, "How China is blocking Tor," *CoRR*, vol. abs/1204.0447, 2012.

[73] Z. Ling, J. Luo, W. Yu, M. Yang, and X. Fu, "Extensive analysis and large-scale empirical evaluation of Tor bridge discovery," in *Proc. IEEE INFOCOM*, 2012, pp. 2381–2389.

[74] M. V. Barbera, V. P. Kemerlis, V. Pappas, and A. Keromytis, "CellFlood: Attacking Tor onion routers on the cheap," in *Proc. ESORICS*, Sep. 2013, pp. 664–681.

[75] R. Jansen, F. Tschorsch, A. Johnson, and B. Scheuermann, "The sniper attack: Anonymously deanonymizing and disabling the Tor network," in *Proc. 21st Annu, Symp. NDSS*, Feb. 2014, pp. 1–15.

[76] C. Egger, J. Schlumberger, C. Kruegel, and G. Vigna, "Practical attacks against the I2P network," in *Proc. 16th Int. Symp. RAID*, Oct. 2013, pp. 432–451.

[77] H. Karaoglu, M. Akgun, M. Gunes, and M. Yuksel, "Multi path considerations for anonymized routing: Challenges and opportunities," in *Proc. 5th Int. Conf. NTMS*, May 2012, pp. 1–5.

Esra Erdin received the B.S. degree in computer engineering from Middle East Technical University, Ankara, Turkey, in 2010 and the M.S. degree in computer science and engineering from the University of Nevada, Reno, NV, USA, in 2012. She is currently working toward the Ph.D. degree with the University of Nevada, focusing on the understanding of social network dynamics and development of phone-to-phone decentralized social networks to address privacy concerns.

Chris Zachor received the B.S. degree in computer science from Central Washington University, Ellensburg, WA, USA, and the M.S. degree in computer science and engineering from the University of Nevada, Reno, NV, USA. Upon receiving the M.S. degree, he accepted a job offer from Great Basin Data Recovery, where he worked as a Digital Forensic Analyst. There, he applied his expertise to a wide variety of investigations ranging from civil litigation to intrusion response. He is currently a Malware Analyst for Kaspersky Lab, Woburn, MA, USA, at their U.S. Virus Lab.

Mehmet Hadi Gunes received the B.S. degrees in computer science and engineering and electronics engineering from Işik University, Istanbul, Turkey, in 2002, the M.S. degree in computer science and engineering from Southern Methodist University, Dallas, TX, USA, in 2004, and the Ph.D. degree in computer science from the University of Texas at Dallas, Richardson, TX , USA in 2008. He is currently an Associate Professor at the University of Nevada, Reno, NV, USA. His research interests include communications (network protocols, health systems, and smart grid communications), complex networks (biological networks, social networks, information networks, graph data mining, and network visualization); Internet measurements (Internet topology, Internet modeling, network sampling, and synthetic graph generation), and network security (anonymizer technologies, communication privacy, and secure cloud, smart grid security). So far, his research has been funded by the Department of Defense, the National Institute of Justice, and the National Science Foundation.