Strengthening Forensic Investigations of Child Pornography on P2P Networks

Marc Liberatore◦ Brian Neil Levine◦ Clay Shields△
◦Dept. of Computer Science, Univ. of Massachusetts Amherst, {*liberato,brian*}*@cs.umass.edu*
△ Dept. of Computer Science, Georgetown Univ., *clay@cs.georgetown.edu*

ABSTRACT

Measurements of the Internet for law enforcement purposes must be forensically valid. We examine the problems inherent in using various network- and application-level identifiers in the context of forensic measurement, as exemplified in the policing of peer-to-peer file sharing networks for sexually exploitative imagery of children (child pornography). First, we present a five-month measurement performed in the law enforcement context. We then show how the identifiers in these measurements can be unreliable, and propose the tagging *of remote machines. Our proposed tagging method marks remote machines by providing them with application- or system-level data which is valid, but which covertly has meaning to investigators. This tagging allows investigators to link network observations with physical evidence in a legal, forensically strong, and valid manner. We present a detailed model and analysis of our method, show how tagging can be used in several specific applications, discuss the general applicability of our method, and detail why the tags are strong evidence of criminal intent and participation in a crime.*

1. INTRODUCTION

The most popular resource for the criminal acquisition and distribution of images and video of child pornography is peer-to-peer (p2p) networks, including BitTorrent and Gnutella[1]. Law enforcement (LE) have both an easy and difficult time policing these networks. On the one hand, it is easy to identify millions of IP addresses trafficking in known child pornography (CP), as we demonstrate in Section 3. On the other hand, this success falls short in several ways. IP addresses and application identifiers are the foundation of all current criminal network investigations, yet IP addresses do not distinguish multiple physical machines behind a NAT box. Similarly, it is difficult to link the activities of a single mobile user moving among many IP addresses. NAT and mobile users represent growing trends.

The value of evidence is the critical difference between forensics and related security research in incident response and privacy; moreover, methods and legal procedures for collecting data differentiate network forensics from simple network measurement. Making guesses or inferences may be suitable for discovering the limits of privacy or advancing incident response, and it may generate an investigative lead, but on its own it will not advance a legal case. IP addresses are an excellent example of the low-value evidence that is in standard use by criminal forensic practitioners — a recent, scathing report by the National Academy of Sciences [18] calls for a scientific overhaul of forensics, including digital forensics.

In this paper, we introduce new techniques that draw a bright line between the measurement or surveillance of these networks and collection of forensically valid evidence from them. Validating the evidence collected during a network investigation is difficult because remote users do not maintain a unique and unmodifiable identifier that can be recovered upon seizure of their machine with a warrant. We propose a novel method of subtly tagging a remote computer over the network to create such an identifier. Our approach is an advance over previous methods of gathering information about a remote computer that rely on statistical characterizations, including clock skew [13] or radiometrics [1]. These past characterizations vary with environmental factors such as temperature or attack [4], leading to both false positives and false negatives, and crucially, lack the ability to link together sequential observation by independent observers. Moreover, we detail why our

[1]Past studies have found that 28% of possessors of child pornography had images of children younger than 3 years old; and that 16% of investigations of CP possession ended with discovery of persons who directly victimized children [22].

ACM CoNEXT 2010, November 30–December 3 2010, Philadelphia, USA.

approach, which is akin to marking bills, is legal.

For this work, we built a system to gather evidence of possession of child pornography on a p2p network. It is in use by law enforcement in 49 U.S. states who have gathered data for us over a five-month period of time. To date, the system and its data have been used to obtain over 1,000 search warrants. We characterize these measurements in order to motivate our tagging techniques. In contrast with methods used today, if our tags were found on a machine during a forensic exam, it would be strong evidence that the machine corresponds to observations of a peer made over the network. Unlike statistical characterization methods, our method has very strong privacy properties: the results can be recovered by investigators only after a search warrant is obtained from a judge. Tags observed by third parties are meaningless. Our careful analysis demonstrates that false positive probabilities can be driven to near zero. The tradeoff is the challenge to make sure tags are retained by the target, to be later discovered during an examination.

Specifically, we make several contributions:

- We present five months of investigations into Internet crime, performed with law enforcement. We show that identifying those trafficking in child pornography on p2p networks is simple, and that such trafficking is unfortunately common, with millions of distinct IP addresses participating.
- We analyze the strength of digital evidence relied on by investigators in these crimes, demonstrating that these techniques on their own are insufficient beyond the standard of probable cause for stationary IP addresses. Moreover, such techniques are insufficient for demonstrating intent and do not work well for mobile users.
- Based on the study, we propose a novel method of strengthening network investigations of criminal activity called tagging. We analyze its design and demonstrate that the chances of false positives can be made insignificant with relatively low overhead.
- Finally, we will show how these tags can be used in several specific applications (including BitTorrent and DNS), discuss the general applicability of our method, and detail why the tags are strong evidence of intent and participation in a crime.

We begin with a statement of the problem, and a description of the relevant attacker models. We then present an empirical analysis of measurements collected on the Gnutella network and among BitTorrent peers, with a focus on the evidentiary value of these measurements. Finally, we present tagging, our proposed mechanism for improving such evidence. This work is a significant extension to our prior work [14, 15].

2. PROBLEM AND ATTACKER MODEL

In this section, we present the motivating problem for our work: network investigations of criminal activity, and forensic validation of the evidence of such crimes. We discuss the investigative process, the legal limitations upon it, and the problem that forensic validation poses. We also present the relevant attacker models. In later sections, we present a set of characterizations that empirically show the scope and importance of the problems we identify here and present a more exact description of our proposed solution.

2.1 Problem Statement

When investigating Internet crimes such as trafficking in child pornography on p2p networks, the general approach of law enforcement is as follows. An investigator issues queries for likely child pornography and gathers results. Some results are chosen for further investigation, and the investigator uses the court system to compel an ISP to reveal a physical location that corresponds to the likely source of network traffic that provided the query results. Under a warrant, the location is searched, any computer systems and media are seized, and the media are examined for evidence of the possession or distribution of CP. We describe the various legal restrictions that US investigators operate under in Section 2.2; these restrictions influence our design decisions.

Our interest lies in effectively identifying the correct end system. In particular, can investigators strongly link network measurements with user behavior and intent? Our goals are twofold: First, we aim to evaluate the quality of the procedures currently used to perform these measurements. We present results of our evaluation in Section 3; in summary, we show that the current procedure of using IP addresses and certain protocol-specific identifiers can fail to identify a unique system in many circumstances.

Thus, our second goal is to improve the quality of evidence and the range of tools available to investigators. In particular, we propose the use of *tagging*. The general mechanism of tagging is to insert bit patterns that are unique to each observation, which we call *tags*, into stable storage media belonging to a suspect during the course of the network-based investigation. These tags can later be recovered from the storage media following a legal seizure, not unlike marked bills might be recovered after an undercover transaction involving stolen property or illegal drugs. The tags can then be used to both link the observations with the media, and to show a pattern of behavior, and thus intent, on the part of the suspect.

2.2 Legal and Practical Issues

Data collected during a network investigation can be used for two distinct, dependent purposes. First, measurements can establish the *identity* of a suspect.

By identity, we mean a network or application identifier that can ultimately be linked to an individual at a given time and place. Second, measurements can be used to establish *intent* to commit a crime: a user might accidentally download a single CP file, but if they have a large and growing collection over the course of months, it is highly unlikely to be accidental. Discovering *intent* requires consistency of identifiers over time, a property that we observe does not always hold.

There are three key considerations for law enforcement conducting network investigations that relate to, but differ from, those of the typical disinterested researcher:

Evidentiary standards: Information that does not meet an evidentiary standard of either *probable cause* for warrants or *beyond a reasonable doubt* for convictions is merely *reasonable suspicion* (a lead), and it is of lesser value. Information collected about a target isn't strong evidence if it came from a third party (e.g., from one peer about another peer). This distinction between leads and evidence is roughly analogous to the difference between observation studies used to generate hypotheses, and controlled studies used to test them. US courts do not accept evidence that is not observed in *plain view*, acquired through a magistrate-approved search warrant or other valid legal procedures. By design, our technique leaves tags that are recoverable only via a search warrant.

Generally, investigators use network identifiers, such as IP addresses, only as part of obtaining a search warrant. IP addresses are generally regarded as meeting only the standard of probable cause — good enough for a search, but not convincing enough for prosecution on their own. As we show here, IP addresses can vary significantly over time, so this level of skepticism is warranted. Application-level IDs suffer from similar credibility problems.

Intent is part of the crime: Many crimes include intent as a requirement for conviction. Possession of CP is legal when unintentional, such as if it is unknowingly held in a spam folder. Among other indicia [9], multiple attempts to download CP can demonstrate intent, as could a growing collection, or the presence of organized archives. Our proposed technique can be used to demonstrate intent in these cases.

Public-use technology only: *Kyllo v. U.S.*, 533 U.S. 27 (2001) established that prior to obtaining a search warrant, investigators cannot collect information using technology that is not already in "general public use". In practice, this means that law enforcement is limited to working within unmodified protocols; our technique obeys this limitation. In *U.S. v. Gabel*, 2010 WL 3927697 it was ruled that software designed to monitor CP sharing on p2p networks does not violate *Kyllo*. And, marking bills is a technique unchallenged in courts.

Finally, we note that we define *forensically valid* techniques based on the standards set by *Daubert v. Merrell Dow Pharma.* 509 U.S. 579 (1993): they have a known error rate, are based on testable hypotheses, are based on accepted scientific methods, and are peer reviewed.

2.3 Attacker Models and Assumptions

We have two actors in our scenario, and we define assumptions for both. We define the **investigator's attacker model** as follows: An investigator of a given p2p system: *(i)* seeks to identify users of the protocol in possession of, or distributing, child pornography — typically, an IP address within their jurisdiction is the endpoint of the network investigation; *(ii)* must work within the protocol, and cannot rely upon criminal activity or privilege escalation to gather evidence; *(iii)* can consider indirect evidence to generate leads, but must have direct evidence to succeed (i.e., seeks a direct network-level connection to a remote user's system). A **criminal's attacker model** and goals are markedly different. A criminal: *(i)* will actively attempt to acquire new CP; *(ii)* can redistribute and advertise possession of CP; *(iii)* can actively manipulate the protocol, violate laws, or engage in anti-forensics to hide their activities. Clearly, a criminal actively attempting to hide their trail will be harder to catch; we discuss the likelihood and impact of such attempts in Section 4.2.

3. AN EMPIRICAL STUDY OF CRIME AND IDENTIFIERS ON P2P NETWORKS

Our goal in this section is to demonstrate the low evidentiary value of IP addresses and application-level IDs when used on their own in network investigations. This low value is the result of widespread use of DHCP, mobile networking, and the presence of botnets, and we provide some quantification of this problem.

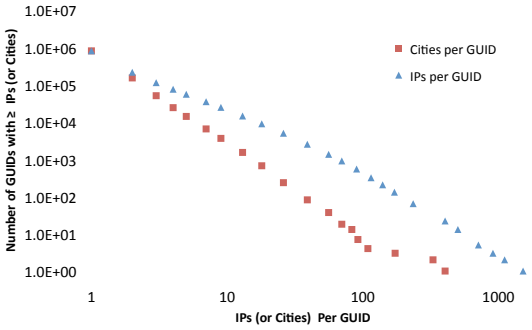
3.1 Collection Methodology

Our empirical results are based on our five-month measurement study of child pornography (CP) file sharing on p2p networks. Our data was collected using a tool we wrote for monitoring and investigating sharing of child pornography on Gnutella networks [15]. As a consequence of our efforts, our tool *RoundUp* has been adopted as a standard for p2p investigations by the US Internet Crimes Against Children (ICAC) Task Force [21]. ICAC is a collection of law enforcement agencies from all 50 states. Data from almost 600 participating detectives' actual investigations of CP trafficking on Gnutella were stored in a centralized system under police control; investigations were not automated. We analyzed an anonymized version of the data.

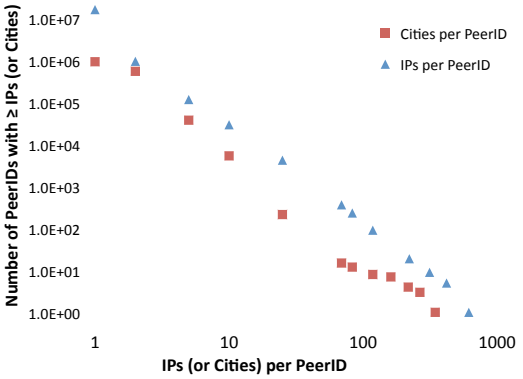
From 10/5/2009 to 3/2/2010, LE using RoundUp collected measurements of 3.07 million IP addresses using 799,556 GUIDs sharing CP in plain view. A GUID is Gnutella's application-level identifier that is chosen at

random during installation, and changeable thereafter. In all, almost 19,000 distinct CP files were observed at least once on the Gnutella network. These CP files are checked manually at least once by law enforcement, and we identify multiple instances by hash value. The records in the database exist due to the particulars of the Gnutella protocol and the efforts of our LE partners. Specifically, records are either the result of a Gnutella *search* for filenames matching CP-related keywords, a direct TCP connection to a remote peer and a *browse* list of their shared files, or *swarming* information from a remote peer that indicates that a third peer has also been sharing the same file (identified by SHA-1). In all cases, these records included remote IP, GUID, software and version, filename, file size, and SHA-1 hash value, and were stored with a timestamp. We used MaxMind, Inc.'s IP-based geolocation service at the time of measurement to place IPs in a city. The database stored information only about peers that shared known or suspected CP.

While Gnutella is not the most popular p2p program, our statistics show it is popular with CP file sharers. We also characterized BitTorrent measurement data collected by Menasche et al. [16]. While our study focused on CP, the data collected by Menasche et al. measured non-contraband content on BitTorrent. Their study measured torrent activity between 8/2008 and 3/2009. Their work includes the details of their measurements, but our focus is on records indicating the IP address and BitTorrent PeerID of participants sharing pieces of torrents. Measurement of these torrents and the peers was performed using PlanetLab-based measurement proxies that gathered information from trackers. For this dataset, all MaxMind queries were performed by us on one day in spring 2010. Our conclusions about evidence on Gnutella GUIDs and IP addresses are validated by observing the same results for BitTorrent PeerIDs and IP addresses, as we describe below. We begin with a summary of the success and limitations of the current investigative approach.



(a) Gnutella



(b) BitTorrent

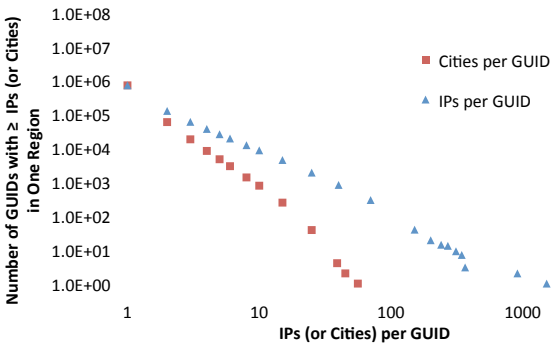
Figure 1: *For a given lower bound on the number of IP addresses (or cities), this plot shows the number of application-level identifiers observed to meet that bound. For example, in the Gnutella data, all observed GUIDs (about 800,000) were observed at one or more IPs, and about 21,000 were observed at 10 or more IPs.*

3.2 The Current Investigative Approach

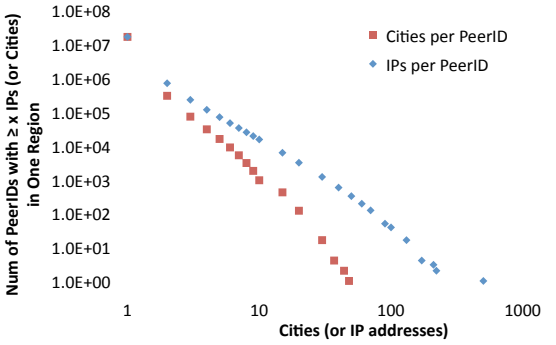
As we stated in Section 2.1, data collected during network investigations are used only as a stepping stone to obtain legal authority to search a physical location for evidence of a crime.

Success of existing approach. In the course of a search, storage media are examined for CP and evidence of intent, such as cached search terms. The presence of this evidence is used to create a case for criminal possession of CP. This methodology has been used successfully in thousands of investigations in the U.S.

Limitations of existing approach. There are three key limitations of the current approach. *(i)* When network investigations lead to search warrants, it is evidence found from the search that is used as the basis for criminal prosecution. In fact, there is often no connection between what is observed on the network and what is found in the search: users may delete files, or install new client software. As a result, it is challenging to prosecute for distribution of CP in the case that some CP is found during a warrant-based search, but it is not the same files that were requested and downloaded by LE from that peer. *(ii)* Positively identifying a seized machine as the same one that was investigated remotely is a challenge. Circumstances such as network address translation, DHCP lease times, and mobile interfaces can cause a mismatch. Similarly, many file sharing applications do not provide a stable unique identifier for the user. For example, BitTorrent does not require fixed PeerIDs, and Gnutella does not ensure each client's self-assigned Globally Unique ID is, in fact, globally unique. *(iii)* Intent is a critical part of the definition of criminal CP possession and distribution. Intent can be demonstrated legally in several ways [9]. Unfortunately, a key form of evidence of intent on p2p networks — sharing on the network over a long period of time — cannot be demonstrated easily in court. An even greater challenge



(a) Gnutella



(b) BitTorrent

Figure 2: *A subset of the data from Figure 1, this plot shows the minimum number of distinct cities or IPs associated with a given application identifier,* limited to the IDs that were observed only in a single small geographic region.

is to definitively show that the same person is responsible for using multiple GUIDs or multiple IPs over time, particularly over open APs in cafes or campuses.

It can be challenging to find the evidence of a crime: The subject of investigation might hide it within the system with encryption or steganography, or might keep the material on a removable storage device that is physically concealed. In cases where the investigator does not locate the material that was seen as being available, it is not clear whether the wrong system was seized or if the material simply hasn't been discovered. A reliable indication that the correct system was seized, as we propose in the Section 4, can help resolve this dilemma and the others above.

3.3 Identity and Intent in P2P Networks

Fig. 1 demonstrates how application IDs can fail as a unique identifier. The figure plots the number of IP addresses associated with each ID in the data. For Gnutella, this consists of GUIDs that have been identified as trafficking in child pornography. In our data, about 21,000 GUIDs were each associated with 10 or more IP addresses. A separate line plots the same data

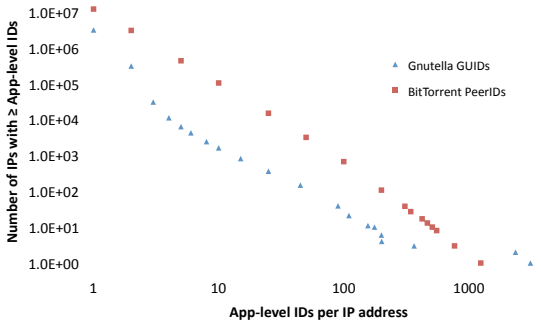


Figure 3: *For a given lower bound on the number of application-level IDs, this plot shows the number of IPs observed to meet that bound. For example, in the BitTorrent data, all observed IPs (about 11,700,000), were observed using one* or more *PeerIDs, while about 102,000 were observed using 10 or more PeerIDs.*

by geographic location, with about 2,800 GUIDs present in 10 or more cities. BitTorrent is similar. For example, there are 5,400 PeerIDs that map to IPs found in more than 10 cities.

One particular GUID was observed in 329 cities around the world using 398 IP addresses. We found this GUID was sharing exactly one file. This file (identified by hash value) was found throughout the network with many different filenames. We assert this GUID is actually a botnet that responds to queries for any term `x` on the network with `x.mpg`, always sharing the same malware content. The existence of this GUID shows the difficulty in assuming that GUIDs are unique identifiers for corroborating an investigation with a seized machine, as this GUID appears to be shared by many users. These observations point to a weakness in such IDs that may skew all data points collected, but in a non-obvious way. Within a legal context, as compared to a network measurement study, the implications are more serious: they limit the value of evidence.

Another problem is posed by mobile users. Fig. 2 isolates IDs that report from 2 or more IP addresses all located one state or region of a country. Since these IDs have IP addresses that map to only one geographic area, we assert that it is most likely one real user, as botnets and misconfigurations are unlikely to be contained to a geographical area. It is unclear to us if these users are sharing their ID with friends, or driving around using open WiFi [19, 20]. This data suggests either that these identifiers are weak, or that users actively move around to avoid detection; either motivates our tagging solution.

Fig. 3 demonstrates the related problem of relying on a peer's IP address as a unique identifier of a specific computer. The figure plots the number of IDs observed per IP address. For example, 6,250 IP addresses were each linked to at least 5 different GUIDs in our database. It is not clear to us whether the GUIDs represent five or more different users behind one NAT box or if one

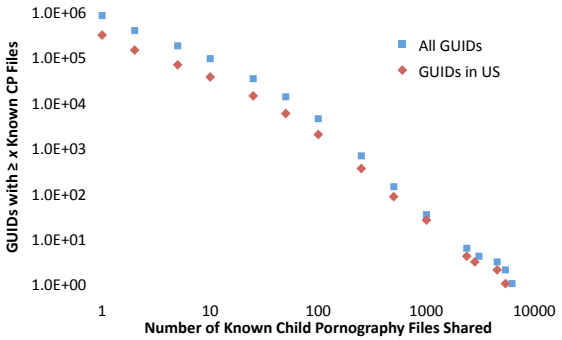


Figure 4: *For a given lower bound on the number of known CP files shared by a Gnutella peer, this plot shows the number of GUIDs observed to meet that bound. E.g., across all observations, all GUIDs (about 800,000) were observed sharing at least one files, while about 90,000 were seen sharing 10 or more files. Note that all peers observed were sharing at least one such item due to LE data collection methodology.*

Figure 5: *For a given lower bound on the number of days a Gnutella peer was seen online, this plot shows the fraction of peers as distinguished by GUID observed to meet that bound (the CCDF).*

user is responsible for all activity from that IP using five or more GUIDs. In BitTorrent, this problem is worse, with 426,425 IPs using at leave five PeerIDs in our study. Since PeerIDs can be generated per torrent, it is almost impossible to link the download of a particular torrent with a specific installation on a computer, or even a specific user on a single computer.

Fig. 4 shows that GUIDs originating in the US are only about 37% of all traffickers. Over 10,000 GUIDs worldwide had 10 or more CP files, and most shared a single file; the metric underscores the scope of the problem as it represents only *known* CP. The users corresponding to these GUIDs often share as-yet-unknown CP or files that have yet to be manually checked by LE, have archived large collections that aren't being shared but found upon execution of a search warrant, or turn out to be contact offenders. Fig. 5 demonstrates that many users are observed repeatedly over long periods of time. The figure also shows that users with larger collections (some have thousands of files) tend to stay on the network longer, an indicia of intent. Unfortunately, with current techniques it is unknown if some of the GUIDs observed once in these two figures are actually the same user.

4. REMOTE DEVICE TAGGING

Measurements that attempt to tie observations together using information provided by an application-level protocol may be flawed. Identifiers turn out to be neither unique nor consistent in all cases. We propose a novel mechanism to remotely tag a device that is under investigation. We begin by presenting the tagging process and follow with a discussion and analytical model of the process. We then show several tagging opportunities that exist in BitTorrent software such as Vuze, and in DNS. We end with a discussion of the increase in investigative power that tagging provides.

4.1 The Tagging Process

We propose the tagging of remote machines by investigators, to leave a record of an observation on the remote machine for later recovery during warranted search. We envision the general process in three steps as follows.

First, investigators discover a *vector* for tags: we define a vector as a set of bits embedded in a protocol that can be set within the bounds of the protocol by the investigator and sent to a remote machine under investigation. Further, these bits or some determinate function of them must be stored by the remote machine on non-volatile media. For example, as detailed in Section 4.4, BitTorrent peers will ask each other their application name and peer ID, and there are minimal restrictions on these values; these values may be stored in a file at the target. These fields can function as tags to uniquely identify the remote machine. No unauthorized access to the target's machine is required; tags are inserted in the normal function of a system.

Second, when directly connecting to a remote machine during an investigation, investigators use an appropriate vector to tag the machine. Tags are selected in such a way that their meaning is not obvious and to minimize the likelihood of collision. The investigator records the tags used to so that they can be validated when recovered. One method of selecting tags is to take a hash value of text representing specific details of the investigation. This hash can be provided as part of the search warrant request to a judge to commit the investigator to one or more values. Law enforcement organizations could release publicly the root of a Merkle tree of all hash values used for a specific time period.

Finally, upon issuance of a warrant by a magistrate, investigators seize a machine and look for known tags on it. These tags may be found in the expected place, or recovery may require more advanced forensic techniques such as file carving. Tags that are recovered from a

seized machine validate that it is a specific system that was investigated over the network. Because recovery requires a magistrate-approved warrant, and because the meaning of the tags is hidden, our approach has robust privacy properties.

There are two ways in which retrieving tags from a machine can fail. False negatives occur when tags that were placed by an investigator are unrecoverable, due to deliberate user action, log rotation, cache eviction, and so on. In these cases, the tags will not be available as evidence. False positives occur when investigators recover tags that they did not actually place — in essence, they recover incorrect evidence. We examine these problems in sequence below.

4.2 Impact of False Negatives

Given that in our model we allow the criminal to erase evidence from their own machine, why do we expect our techniques to work at all? There are several answers. First, unlike most mechanisms in security, most forensic mechanisms are not subject to *catastrophic failure*: even if one person can and does erase evidence, that does not imply that everyone will act accordingly, nor does it mean that one person can erase evidence for everyone else. Further, it is still worth investigating those that do not erase evidence. In contrast, if there exists a security exploit in Windows, then one user can comprise every Internet-accessible Windows machine.

Secondly, these crimes are not committed by persons with great savvy — the quantitative proof is the measurement we present in Section 3: we identified 3,069,628 IPs (using 799,556 GUIDs) sharing known images of confirmed child pornography. These observations are based on a database of hash values. Anyone can trivially circumvent the hash match, yet millions did not.

Thirdly, we expect that generally application developers will not help unsavvy criminals nor aim to thwart tagging mechanisms. Since the tags are impossible to trace back to investigators, developers will have no sense of whether they are being used. To be sure they are not open to any tagging and not just the mechanisms we propose here, developers would need to perform a covert channel analysis [7] on their program as well as all OS libraries in use, likely blocking all caching and similar output from both. Moreover, our methods are designed to tag system mechanisms that improve performance when left enabled (e.g., the DNS cache); we assert that developers are in general more interested in improving performance than protecting traders of child sexual exploitation imagery. Of course, the copyright enforcement actions of trade groups such as the RIAA and MPAA are thwarted actively by some p2p developers. However, the civil torts these groups pursue require only *relevance* for subpoena of a target machine (the 4th Amendment does not apply) and the much lower standard of a *preponderance of evidence* at trial. In short, our tagging techniques would be overkill for supporting civil torts. Again, there is no incentive to remove tagging vectors.

4.3 Modeling False Positives

Tags are most useful as evidence after a search warrant has been executed. Therefore, how certain are tags as evidence? In other words, what is their false positive (FP) rate? How do different tag sizes, numbers of tags, and tagging schemes interact with the FP rate? We probe these questions here.

We define false positives as when a machine that was never tagged appears to be tagged. The analysis we carry out to determine false positives applies equally to the scenario where an adversary places tags on a third-party victim's machine in an attempt to frame the victim, as we assume the attacker doesn't know which tags are used by investigators.

Model assumptions. Assume investigators tag target machines with an n-bit tag each time they are observed on the network (called a *session*), and they keep a database of T entries. The number of entries is exactly the space of all tags that have ever been or will ever be assigned for a distinct taggable event. Each entry will include other essential information about the investigation: the name of the investigator, the date, the tagged IP address, etc. Here we set $T = 2^{\frac{n}{f}}$, and therefore the chance that a recovered tag (that was not placed by investigators) is a false positive is $T/2^n$. We assume $f > 1$, where $1/f$ is the fraction of table space used for a tag, since when $f = 1$ the chance of a false positive is 1. We discuss how f affects performance below, and in fact it is one of two variables that must be decided ahead of time. We let L be the number of candidate tags that are discovered.

In the analysis below, we assume a log file is recovered from a seized machine and that, unbeknownst to investigators, the machine has never been tagged. In other words, any bit fields that contain apparently valid tags contain bit strings drawn from some unknown distribution, which we assume is independent of the tagging database. For simplicity, we assume that distribution to be uniform.

Large tags. The simple case for tagging is when n is very large; in that case, it is easy to make it improbable that a tag found on seized machine falsely matches a tag in the database. The chances that one or more of L candidate tags match stored values in the database is

$$\begin{aligned} Pr\{\text{False positive}\} &= 1 - Pr\{\text{no matches}\} \\ &= 1 - (1 - \frac{2^{n/f}}{2^n})^L \end{aligned} \quad (1)$$

However, the maximum value of n is not chosen by the investigator; it is a constraint of the tagging vector, as

discussed in Section 4.4. For example, if $L = 2000$ and $n \leq 32$, the chances of a false positive is greater than 3%, which is most likely too high.

Small tags. In some situations, the tag size n is limited and we require a low false positive rate. To overcome this limitation, we have the investigator generate and use many *subtags* per session. Subtags are generated by splitting n-bit tags into k equal-length parts. An investigator then subtags a remote machine k times in a session.

There are two scenarios that we must consider.

- **Case A.** The subtags are stored on the target machine in a *preserved order* that can be recovered.
- **Case B.** The subtags are stored on the target machine in an *unordered set* that prevents ordered recovery. For the unordered case, we offer two solutions: (B1) tagging the target k times each session; and (B2) allocating space in the subtags to denote the order for recovery of the full tag, which we show below is a better solution.

We derive the false positive rates of the three approaches below and then compare their performance.

Case A: Order Preserved: Concatenated subtags. In this case, we assume subtags are written to a sequential log file, and that investigators can reconstruct the original tag by assembling subtags in the order they are recovered from the log file. It may be that other data is inserted into the target's log between subtags, which can result in false positives. Here, we model the most conservative case: we show the number of false positives given that none of the L candidate subtags were placed by investigators.

When the machine is recovered, the investigator will accept the machine as tagged only if k of the L subtags, when concatenated, appear in the database of T assigned tags. The problem is that investigators must try every combination of $\binom{L}{k}$ found, which creates a large number of potential false positives. Here, $T = 2^{\frac{n}{f}}$ as before (and subtags are n/k-bits long). The false positive rate of the concatenated tags is

$$\begin{aligned} Pr\{\text{False positive}\} &= 1 - Pr\{\text{no full tag matches}\} \\ &\leq 1 - \left(1 - \binom{L}{k}\frac{1}{2^n}\right)^{2^{\frac{n}{f}}} \end{aligned} \quad (2)$$

Note that this is a conservative upper bound, not an equality, as we have elided the inclusion-exclusion terms.

Case B1: Unpreserved Order: Multiple subtags. In this case, the target machine does not store tags in a preserved order. In this solution to the problem, we have investigators tag the machine with k subtags that share the same database. Therefore, there is a limit of $T = 2^{\frac{2}{fk}}$ tags that can be assigned.

The false positive rate for the case of k subtags of $\lfloor n/k \rfloor$-bits each is given by a Binomial distribution.

$$\begin{aligned} Pr\{\text{F.P.}\} &= Pr\{k \text{ or more of } L \text{ subtags match}\} \\ &= 1 - \sum_{i=0}^{k-1} \binom{L}{i} (2^{\frac{n}{fk} - \frac{n}{k}})^i (1 - (2^{\frac{n}{fk} - \frac{n}{k}}))^{L-i} \end{aligned} \quad (3)$$

Eq. 3 quantifies the tradeoff between using one tag of n bits and k subtags of one bit each, and all cases in between.

Case B2: Unpreserved Order: Labeled subtags. When the tagged machine stores the subtags in an unordered set, a better solution is to give each subtag its own database, or to otherwise label each subtag to constrain the set of possible databases it could be in. For example, we can reserve $\log_2 k$ bits in each subtag to denote which database it is in. In that case, each subtag has length $r = \lfloor n/k \rfloor - \lceil \log_2 k \rceil$ bits, and we have $T = 2^{rk/f}$ tags possible. To determine the false positive probability, we assume that the L candidate tags are equally divided among the k subtag databases. Therefore, there are $(L/k)^k$ candidate full tags to evaluate; each must not match an assigned value.

In this case, the false positive probability is modification of Eq. 1:

$$\begin{aligned} Pr\{\text{F.P.}\} &= 1 - Pr\{\text{none of } (\tfrac{L}{k})^k \text{ subtags match}\} \\ &= 1 - \left(1 - \frac{2^{rk/f}}{2^{rk}}\right)^{(\frac{L}{k})^k} \end{aligned} \quad (4)$$

As we discuss in Section 4.4.2, specific ranges of IP addresses such as CIDR blocks can be used as tags. In this case, the fixed prefix of the CIDR block serves in place of the $\log_2 k$ bits that would otherwise be reserved to denote the sub-database. The no-cost nature of these prefix bits explains the improved performance of this method in the comparison below.

4.3.1 Sessions Taggable by Each Method

To compare these three methods, we assume the investigator knows the vector-specific length of each subtag and value of L, and has a desired FP rate. Her job is to select f and k such that the false positive rate is achieved and the number of sessions that can be tagged is maximized. In general, larger values of f and k lower the FP rate but reduce the number of sessions that can be tagged. We assume the goal is to minimize k, since if more tags are required to be stored, it is more likely that in general that the tags may be removed by normal operation of the machine.

Accordingly, we evaluate three questions: *(i)* For a desired FP rate ρ, what is the minimal value of f? *(ii)* How does the number of sessions, T, that can be tagged vary with f? *(iii)* What is an acceptable FP rate? We explore these questions in several ways.

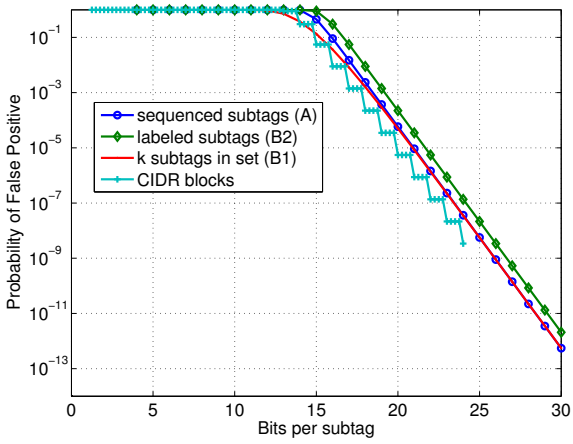


Figure 6: *The false positive rate as a function of the number of bits per subtag, corresponding to $k = 4, f = 3$, and $L = 2000$. The exact values plotted are from on Equations 2, 3, and 4.*

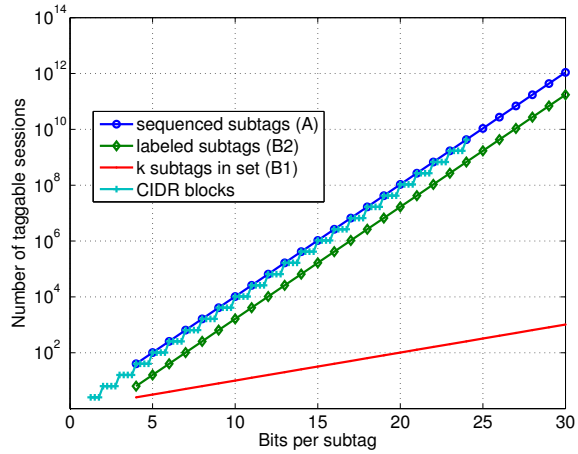


Figure 7: *The number of sessions that can be tagged as a function of the number of bits per subtag corresponding to $k = 4, f = 3$, and $L = 2000$.*

First, as quantitative examples, we compute the FP rate ρ and number of sessions for each method when $k = 4, f = 3$, and $L = 2000$ based on Equations 2, 3, and 4. The false positive rates for the three techniques is shown in Fig. 6. Note the x-axis is the subtag size and not n. In all cases, the probability of a false positive decreases exponentially with the subtag size, and it is less than 10^{-6} when subtags are at least 22 bits. Similarly, Fig. 7 compares the number of sessions that can be tagged by each method, including the CIDR variant of Case B2, based on the definitions of T for each. The number of sessions offered by Case B1 is many orders of magnitude lower than the other solutions.

Second, we address the broader question of how to choose f and T for a given FP rate ρ. While Figs. 6 and 7 used the subtag size as the independent variable, here we assume the subtag size is fixed and that L is given.

For Case A, we solve for a minimal value of f from Eq. 2 as

$$f = n/\log_2\left(\log(1-\rho)/\log\left(1-\frac{\binom{L}{k}}{2^n}\right)\right) \quad (5)$$

Since $T = 2^{n/f}$, we can state T in terms of ρ as

$$T = \log(1-\rho)/\log\left(1-\frac{\binom{L}{k}}{2^n}\right) \quad (6)$$

For Case B2, we have from Eq. 4

$$f = rk/\left(\log_2(1-(1-\rho)^{1/(\frac{L}{k})^k}) + rk\right) \quad (7)$$

and since $T = 2^{rk/f}$, we can state T in terms of ρ as

$$T = (1-(1-\rho)^{1/(\frac{L}{k})^k})2^{rk} \quad (8)$$

Second, we offer the following simple algorithm that allows an investigator to set f and k. (1) Select a desired false positive rate ρ, and set $k = 1$, which determines the subtag length. (2) Calculate f and the maximum number of taggable sessions, T, using either Eqs. 5 and 6 (Case A) or Eqs. 7 and 8 (Case B2). (3) If T is too small, increase k (which implies an increase to n), and goto Step 2.

The question remains as to what FP rate is appropriate. Historically, law enforcement around the US have made about $A = 2000$ arrests per year. We use this number as an example, however in practice, we can keep one set of tables per available tagging channel. To set ρ, we assume that all the arrests are mistakes, and let $\rho = 0.1A$ such that the expected number of arrests that have a false positive tag is 0.1. If a more conservative estimate is desired, Chernoff bounds can be used to ensure that values above the expected mean occur with very low probability.

4.4 Available Tagging Vectors

For tagging to be practicable and of maximum value, several conditions must hold. First, a machine under investigation must be able to receive data from a remote source. Second, that data must be stored in a fashion that can be retrieved by investigators if the machine is physically seized. Third, investigators must be able to manipulate this data in such a way that it is specific to a single investigation — re-using tags dilutes or nullifies their evidentiary value, violating the assumptions we make in our security analysis. The information will only be recovered when legal authorization by means of a search warrant so allows. Here, we discuss the general manner by which such opportunities can be found and present several specific tagging vectors.

4.4.1 Discovering available vectors

In our experience, tags ultimately reside in one of two places: log (or audit) files, and cache files. Across many

systems, log files record both regular and exceptional events. Log files exist for audit and debugging reasons, and typically include many details of triggering events. Cache files exist to improve performance or reliability of systems. For example, most p2p applications store data about remote peers for several purposes: to allow for decentralized operation and bootstrapping; to enable efficient load distribution; or to enable optimizations such as tit-for-tat. In such cases, the removal of the taggable files would be detrimental to the performance of the system that creates them, and therefore these are the best candidates, as we discuss later.

There are several ways in which tagging opportunities can be discovered. We used a manual, ad hoc process to discover the tagging opportunities we describe below. We conjecture that both static and dynamic analysis techniques can be applied to applications to find tagging vectors and we believe this is an interesting problem for future work.

4.4.2 *Specific tagging vectors*

Here we give three specific examples of vectors that currently exist in the regular functioning of p2p file sharing software, as well as mentioning other possible opportunities. We present these opportunities as proofs-of-concept, and not as fully developed tagging systems. A limitation of our work is that we do not evaluate the churn of these systems quantitatively.

Peer caches. In BitTorrent, peers may actively download and upload a torrent for long periods of time. Files are large, and the culture of BitTorrent users is such that continuing to provide upload bandwidth for a torrent is encouraged, while disconnecting immediately after finishing your own download, possibly while throttling your own uploads is discouraged. To maintain state for these active torrents across application restarts and machine power-offs or suspensions, most BitTorrent clients write relevant information to a cache file of some sort. Were this caching disabled the performance of BitTorrent would be worse, as clients would have to re-discover all peers after application restarts. As we discussed in Section 2.3, removing this functionality from the program is a poor defense.

The μTorrent client stores, per user-account and per torrent, the IP addresses and ports of remote peers sharing that torrent. (The same client, rebranded, is also the BitTorrent client distributed by BitTorrent, Inc.) These IPs and ports are stored in a file named `resume.dat`, which is a bencoded[2] dictionary (associative array). This dictionary is keyed by the each active torrent's infohash. An infohash uniquely identifies the content of a torrent. Each value associated with a torrent's infohash is another dictionary. In this dictionary, the key "peers6" encodes 128-bit IPv6 addresses and 16-bit ports of peers. IPv4 addresses are encoded as backwards-compatible IPv6 addresses. Crucially, these addresses and ports can be provided not just from the tracker, but from other peers through the peer-exchange extensions to the BitTorrent protocol. In our observations, these values need not represent reachable or even valid peers to be entered into the peer cache, presumably because well-behaved BitTorrent clients may ignore incoming connections.

In a similar fashion, Vuze stores, per user-account, a cache file for each active torrent, named `<infohash>.dat`. These bencoded dictionaries contain a key explicitly describing the "tracker_cache", including entries for "tracker_peers" formatted as follows:

```
[  {  'ip': '83.253.52.14',
      'port': 6886,
      'prot': 1,
      'src': 'Tracker'},
   {  'ip': '87.7.101.196',
      'port': 54650,
      'prot': 1,
      'src': 'PeerExchange'}, ...
]
```

Here, even the source of the remote peer is listed, so we can exclude all non-"PeerExchanged" addresses from consideration when recovering tags, eliminating a potential source of false positives.

In both cases, the address and port serve as a tag, though we have observed that the order of the entries in the peer cache is not preserved in either case. Investigators can use CIDR blocks of address space as tags for IPv4 (e.g., leaving 24-bit subtags for /8 blocks; see Figs. 6 and 7) and equivalent mechanisms in IPv6. Investigators can rent small blocks of address space from many different ISPs to prevent any particular address range from appearing as obviously enforcement-related.

As currently implemented, neither peer caching mechanism requires the investigator to answer BitTorrent protocol messages sent to the addresses that may be stored, so the investigator can insert tags chosen from the IPv6 or v4 address space as appropriate through the peer exchange mechanism. If the implementation were changed to require these remote tags to be valid, the investigator could limit the tags to addresses under their control, running appropriately modified p2p software.

DNS cache entries. By default, μTorrent performs reverse DNS lookups on peer IPs once it has connected to them, and Vuze can be configured to do likewise. Many p2p applications include this feature. By performing this lookup, a p2p application likely causes the host OS to cache the returned DNS entry (supplied by LE with unique values). The existence of this entry, and possibly the textual value of the DNS entry itself, serve as a tag.

Other tagging opportunities. Depending upon the p2p system and implementation, there are other targets of opportunity. An obvious potential target is the pay-

[2]Bencoding is a data serialization technique specified by the BitTorrent protocol.

load data being transmitted by the p2p users. Most systems use some sort of hashing scheme to prevent the deliberate poisoning of exchanges with bad data. Still, if this data is ever written to disk, for example as either as a temporary file or through the VM system, traces of it may persist and be recoverable through standard forensic means. Relative to the tag sizes derived earlier, the immense size of even a relatively small file system allocation unit could provide a definitive tagging opportunity.

Vuze log files. Vuze (formerly Azureus) is among the most popular of BitTorrent clients. It creates user-account-specific log files for several purposes, including debugging. One such log file, named `debug.log` (or a variation thereof, when rotated), contains at least two obvious candidates for tagging.

The first arises from the evolving nature of the BitTorrent protocol specification. In particular, the format by which BitTorrent peers are identified, the *peerID*, is not fully specified. To aid developers in discovering and naming new BitTorrent implementations, peerIDs in unrecognized formats are saved to the log. As these peerIDs can be arbitrarily chosen 120-bit strings, they present an ideal tagging target. Similarly, when peers give longer-form identifiers, as permitted in both the LibTorrent Extension Protocol and the Azureus Messaging Protocol, unknown or mismatched identifiers are written to the log file. Below is an example real entry, demonstrating a large number of available tagging bits:

```
- [2009] Log File Opened for Vuze 4.2.0.2
- [0406 09:16:22] unknown_client [LTEP]:
        "Unknown KG/2.2.2.0" / "KGet/2.2.2"
        [4B4765742F322E322E32],
        Peer ID: 2D4B47323232302D494775533761494E45425245
- [0406 09:22:14] mismatch_id [LTEP]:
        "BitTorrent SDK 2.0.0.0" / "BitTorrent SDK 2.0"
        [426974546F7272656E742053444B20322E30],
        Peer ID: 2D4245323030302D2759514731415950276462 62
```

The second tagging opportunity arises from a more subtle side channel present in the log. In particular events for protocol errors can be triggered at timed intervals, such that the inter-event timing forms a side channel. As a simple example, one second between log entries could represent an encoded `0`, and two second delays a `1`. Our past work has shown that channels of this kind are not difficult to implement and that data can be encoded and sent reliably even in the absence of feedback about the time on the receiving system [2]. This is made easier by the fact that BitTorrent files are often large and clients tend to stay on the network for a long period of time [10].

In both cases, these events are logged, but no information is given to the user through the GUI. Both cases clearly allow for tagging, as the log includes information chosen by the remote peer. Further, these tags preserve sequencing information, due to their explicit timestamps.

4.5 Tagging to Improve Forensic Investigation

While tagging can be used by one investigator to later identify a particular system, it becomes a more powerful tool for law enforcement when used collaboratively by different investigators. By providing a central repository of tags placed on systems along with a history of when and why the system was tagged, tagging can support a broader and more effective investigation. This approach mirrors the very successful database currently provided by our RoundUp tool, which records when a particular user, identified by IP address or p2p client GUID, was seen sharing contraband in plain view.

Using the tag database, investigators who seize a system can retrieve tags from the system and use them to identify other times the system was seen and tagged online. This can help identify the same system that was seen with different IP addresses or that was participating in different p2p networks. The ability to recognize the same system in different contexts provides new enforcement opportunities currently unavailable to LE. In particular, it will allow LE to better understand the proclivities of the owner of the system and allow better understanding of the overall community of offenders. For example, if a system is found that has been tagged while sharing contraband frequently or over different networks, the user may be more likely to be a serious offender than the owner of a system that has fewer tags.

The use of tagging can also help law enforcement understand the scope of the overall problem of CP on p2p networks. While it is possible to count how many users are sharing contraband over time, it is not possible to tell which are users that are reappearing and which are new users coming online. The tags found on seized computer will allow better estimates of the problem by enabling methods similar to the mark-and-release method of estimating animal population in the wild.

5. RELATED WORK

Existing methods of matching traffic to end systems rely heavily on mutable identifiers such as IP addresses. Alternatives available to investigators use statistical properties that also are both protean and easily falsified. For example, a remote peer's clock skew can be measured based on TCP headers [13, 17], but this value is both affected by temperature and easily falsified by modifying the TCP header. Even radiometric identifiers [1] can be attacked [4]. To our knowledge, these statistical measures of a remote peer are not used by practitioners because of these problems. Xie et al. [23] also note that the ephemeral assignment of IP addresses presents accountability problems in network security and incident response. In contrast, our focus is on law enforcement who would be unable to subpoena evidence from a myriad ISPs and web sites to deploy Xie et al's solution. Moreover, our approach has a tunable error

rate. Their approach has the advantage of distinguishing many hosts at once.

Our user-centric approach differs from past measurement and characterization of deployed p2p networks [3, 11, 16]. These past works have focused primarily on performance-related metrics, with an eye toward improving typical user experience in file sharing networks. In contrast, our focus is on forensic investigation and criminal conduct. Finally, our work employs well-known steganographic, watermarking techniques, though we apply them to a novel scenario.

6. CONCLUSION

In this paper, we have made two major contributions. First, we have shown through analysis of large-scale, real-world data sets that both network- and application-level identifiers are not reliable evidence on their own for more than probable cause. Second, we have presented tagging, a mechanisms for the active creation of reliable, verifiable identifiers that easily meet the standard of beyond a reasonable doubt. We have provided a strong analytical model demonstrating the applicability of tagging to this problem, and we have presented several distinct avenues for tagging across applications and the OS.

Our work in forensics exists at the intersection of network measurement and security, and it is informed by our practical experience in this area with law enforcement colleagues. In general, the best method of approaching real problems is often risk mitigation rather than perfect security, and criminal investigation is no exception. Even though evidence such as tags can be erased or lost, catastrophic failure is not present in forensics to the degree that it is in security as no one user can erase all evidence for all users. Moreover, erasing specific data from a machine is much harder than erasing all data, and the latter is still an indication of something. This mode is true for other areas of practical security. For example, despite power monitoring attacks on cryptosystems [6, 12] most people do not use tamper-proof hardware. Although their system is not proof against these attacks, there is still value in defeating most other attackers. Similarly, the Tor privacy network is architected to provide reasonable performance instead of perfect security against known attacks [5]. And the TSA admits it cannot defeat all terrorists, and instead simply mitigates risks [8].

Acknowledgements. We are grateful for the collaboration of Det. Capt. Thomas Kerle (MA State Police) and Corp. Robert Ederly (PA State Police), as well as 600 members of US state and federal law enforcement who helped us carry out our measurements. We thank George Bissias and Ramesh Sitaraman for insightful discussions of the tagging FP rate analysis.

This work was supported in part by NSF awards CNS-0905349, CNS-1018615 and DUE-0830876 and NIJ award r2008-CE-CX-K005. The opinions, findings, conclusions, and recommendations expressed do not necessarily reflect those of the sponsors or ICAC.

7. REFERENCES

[1] V. Brik, S. Banerjee, M. Gruteser, and S. Oh. Wireless Device Identification with Radiometric Signatures. In *Proc. ACM MobiCom*, pages 116–127, October 2008.

[2] S. Cabuk, C. Brodley, and C. Shields. IP Covert Channel Detection. *ACM Trans. on Info. Sys. Sec.*, 12(4):1–29, 2009.

[3] J. Chu, K. Labonte, and B. N. Levine. Availability and Locality Measurements of Peer-to-Peer File Systems. In *Proc. ITCom*, volume SPIE 4868, pages 310–321, July 2002.

[4] B. Danev, H. Luecken, S. Capkun, and K. El Defrawy. Attacks on physical-layer identification. In *Proc. ACM Conf on Wireless network security*, pages 89–98, 2010.

[5] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proc. USENIX Security Symposium*, pages 303–320, Aug. 2004.

[6] T. Eisenbarth et al. On the Power of Power Analysis in the Real World. In *Proc. CRYPTO*, pages 203–220, 2008.

[7] V. Gligor. A guide to understanding covert channel analysis of trusted systems. Technical Report NCSC-TG-030, National Computer Security Center, November 1993.

[8] K. Hawley. TSA's Take on the Atlantic Article. `http://www.tsa.gov/blog/2008/10/tsas-take-on-atlantic-article.html`, Oct 21 2008.

[9] T. Howard. Don't Cache Out Your Case. *Berkeley Technology Law Journal*, 19:1157–1575, 2004.

[10] M. Izal, G. Urvoy-Keller, E. W. Biersack, P. Felber, A. A. Hamra, and L. Garcés-Erice. Dissecting BitTorrent: Five Months in a Torrent's Lifetime. In *Proc. ACM PAM*, 2004.

[11] A. Klemm, C. Lindemann, M. K. Vernon, and O. P. Waldhorst. Characterizing the query behavior in peer-to-peer file sharing systems. In *Proc. ACM IMC*, pages 55–67, 2004.

[12] P. Kocher, J. Jaffe, and B. Jun. Differential Power Analysis: Leaking Secrets. In *Proc. CRYPTO*, pages 388–397, 1999.

[13] T. Kohno, A. Broido, and K. Claffy. Remote physical device fingerprinting. *IEEE Transactions on Dependable and Secure Computing*, 2(2):93–108, April-June 2005.

[14] B. Levine, M. Liberatore, and C. Shields. Challenges to Digital Forensics Posed by Mobile Systems and Ubiquitous Computing. In *Proc. ARO Wkshp on Digital Forensics*, Sept. 2009.

[15] M. Liberatore, R. Erdely, T. Kerle, B. N. Levine, and C. Shields. Forensic Investigation of Peer-to-Peer File Sharing Networks. In *Proc. DFRWS Annual Digital Forensics Research Conference*, August 2010.

[16] D. Menasche, A. Rocha, B. Li, D. Towsley, and A. Venkataramani. Content Availability and Bundling in Swarming Systems. In *ACM CoNext*, pages 121–132, 2009.

[17] S. Murdoch. Hot or Not: Revealing Hidden Services by their Clock Skew. In *Proc. ACM CCS*, pages 27–36, Oct 2006.

[18] National Research Council, Committee on Identifying the Needs of the Forensic Sciences Community. *Strengthening Forensic Science in the United States: A Path Forward*. The National Academies Press, February 2009.

[19] J. Stockwell. WiFi Turns Internet Into Hideout for Criminals. Wash. Post `http://www.washingtonpost.com/wp-dyn/content/article/2007/02/10/AR2007021001457.html`, Feb 11 2007.

[20] U.S. Dept. of Justice. Magic Valley Man Gets 12+ Years For Child Pornography. `http://saltlakecity.fbi.gov/dojpressrel/pressrel08/childporn011108.htm`, Jan 2008.

[21] U.S. Dept. of Justice. The National Strategy for Child Exploitation Prevention and Interdiction: A Report to Congress. `http://www.projectsafechildhood.gov/docs/natstrategyreport.pdf` pages 19–22, August 2010.

[22] J. Wolak, D. Finkelhor, and K. J. Mitchell. Child-Pornography Possessors Arrested in Internet-Related Crimes: Findings From the NJOV Study. Technical report, National Center for Missing & Exploited Children, 2005.

[23] Y. Xie, F. Yu, and M. Abadi. De-Anonymizing the Internet Using Unreliable IDs. In *Proc. ACM SIGCOMM*, pages 75–86, August, 2009.