

# Identifying Mobiles Hiding behind Wireless Routers

Yinjie Chen\*, Zhongli Liu\*, Benyuan Liu\*, Xinwen Fu\* and Wei Zhao<sup>†</sup>

\*University of Massachusetts Lowell, Email: {ychen1, zliu, bliu, xinwenfu}@cs.uml.edu

<sup>†</sup>University of Macau, Macau, China, Email: weizhao@umac.mo

**Abstract**—The network address translation technique (NAT) is widely used in wireless routers. It is a low cost solution to IPv4 address space limitations. However, cyber criminals may abuse NAT and hide behind wireless routers to use mobile devices and conduct crimes. To identify a suspect mobile device, we should be able to map the suspect public traffic on the Internet to the private traffic behind the wireless router in WLAN. In this paper, we propose a suite of novel packet size based traffic marking techniques to identify suspect mobiles in encrypted wireless networks as well as open wireless networks. To cope with severe packet loss during wireless sniffing, we proposed to use error correcting codes to improve detection rate. We conducted extensive analysis and experiments to demonstrate the efficiency and accuracy of our schemes, which achieve high detection rate and very small false positive rate. The proposed strategies can be used for law enforcement for combatting cyber crimes in wireless network crime scene investigations.

## I. INTRODUCTION

Mobile computing, wireless networks, and the Internet have become convergent, pervasive and ubiquitous. WiFi routers are sharing a large share of this booming market because they provide tetherless communication, and are easy to use and cheap [1]. A wireless router often uses NAT (Network address translation) to provide its users a shared public IP to the Internet. A DHCP server often running on the wireless router dynamically assigns private IPs to its clients.

However, the pervasive deployment of WiFi has provided an easy venue for cyber criminals to commit crimes including accessing illegal content anonymously. Experienced hackers don't hack from home without any cover. It is too easy for law enforcement to trace an IP address and subpoena the ISP for the location. The notorious hacker, Max Butler, who was sentenced to 13 years in prison [2] in February 2010, often stayed at a large hotel in downtown San Francisco and used open wireless networks or hacked wireless networks with weak encryption to commit remote attacks while hiding behind wireless routers [3]. We expect that similar strategies are used by many such tech savvy cyber criminals given widely available open wireless networks and wireless security cracking tools including Aircrack [4]. Many household and small business wireless routers don't maintain logs or mechanisms for such wireless network forensic investigations [1].

To identify the criminal mobile device hiding behind encrypted wireless routers, law enforcement needs to correlate suspect public traffic on the Internet and private wireless traffic in the WLAN. The suspect public traffic can be network attacking traffic or child pornography downloading traffic that have been identified by intrusion detection systems and Internet surveillance tools. Once the private wireless traffic

and the mobile MAC have been identified, further approaches such as 3DLoc [5] can be applied to locate the suspect for search warrant from courts.

In this paper, we propose a suite of novel packet size-based traceback approaches to identify a wireless mobile device which hides behind encrypted wireless networks. We manipulate the suspect traffic packets (such as identified attack traffic or those from suspicious foreign sites) on the Internet in order to embed secret watermarks into the packet flow. The IP of the suspect traffic flow corresponds to the public IP of the target wireless router, which can be located through tools such as *geoIP* [6]. Therefore, a law enforcement around the target wireless networks can sniff the traffic and recognize the secret watermarks. These approaches can be used in unencrypted wireless networks. We also discussed much easier traceback approaches for simpler unencrypted wireless crime scene investigation.

Our contributions can be summarized as follows:

(i) We proposed to identify cyber criminals hiding in both encrypted and unencrypted wireless networks. The problem is novel and law enforcement has urgent demands of these wireless network crime scene investigation techniques [7].

(ii) To combat severe wireless packet loss during sniffing, we proposed to use error correcting codes including *repetition code*, *Hamming code* and *online code* and recover lost watermarks. Since criminal mobiles often hide in buildings and law enforcement sniffs outside, wireless packet loss at the wireless sniffer is common because of collision and various fading [8]. To recover the encoded messages embedded in target wireless traffic from dynamic background wireless traffic, we propose advanced watermark detection algorithms which achieve high detection rate and small false positive rate. We present a comprehensive set of theoretical analysis and experimental results to demonstrate the effectiveness of our packet size-based traceback approaches with error correcting codes.

(iii) We proposed a multi-channel wireless sniffer utilizing AirPcap dongles. Since we don't know which wireless channel the suspect communicates on, such a multi-channel sniffer is necessary in generic wireless network crime scene investigations. This sniffer demonstrated excellent performance in acquiring and processing massive wireless traffic.

The rest of this paper is organized as follows. Section II introduces our basic idea of traceback through a NAT router. Section III introduces the traceback approach for encrypted wireless network. The evaluation analysis is presented in Section V. Section VI surveys related literature on traceback mechanism. At last, Section VII will conclude this paper.

## II. BACKGROUND AND PROBLEM DEFINITION

In this section, we first define the problem of tracing 802.11-compliant mobile devices as per the need of law enforcement officers. We also present the basic idea in our approach, and a description of our target application. Then we discuss about the performance metrics for our approach.

### A. Background

With the expansion of open-access and easy crackable WiFi networks, cyber criminals now may utilize such networks existing at various places such as hotels to conduct anonymous criminal activities. The network address translation technique (NAT) is been widely used at home and small business wireless routers today. It is implemented in various routing devices that enable multiple hosts with different private IP addresses connecting to Internet through one public IP address. This mechanism creates difficulties for the law enforcement to identify the cyber criminals, who are accessing illegal content such as child pornography at hidden or foreign sites.

Although law enforcement can track the public network traffic to the wireless router public IP and send query to the Internet service provider (ISP) for geolocation information of any public IP address [6], there is no efficient method to determine which mobile behind the wireless router is the target device, particularly when the wireless router is shared by many users and applies link layer encryption. Recall that most of those wireless routers are not equipped or enabled with necessary forensic capabilities [1] to identify the private IP and MAC connecting to the public IP.

### B. Traceback in Wireless Networks

Figure 1 shows the overall watermarking traceback model. The criminal uses a mobile device, denoted as target device, which is connected to a wireless router. The criminal is downloading illegal content from a web server, denoted as *illegal server*. Assume that the *law enforcement* detects such illegal downloading and derives the geolocation of the wireless router through a geoip service based on the ip address in the packet header. Assume that law enforcement is able to manipulate the target traffic packet size and embeds watermarks into the target traffic. The next step for the law enforcement is to sniff wireless traffic around that geolocation of the wireless router's IP in order to correlate private wireless traffic flows to the suspect public network flow based on the watermarks.

We consider two cases of wireless networks: encrypted and unencrypted.

**Unencrypted Wireless Network** If the target device is associated to an unencrypted wireless network, all packet header information are discernible during traffic sniffing. Luckily, although NAT transforms the private IP into the public IP, it keeps much information such as IP sequence number and illegal server IP and port [9], [10]. Law enforcement can utilize those traffic invariables to correlate flows into and out of the wireless router and infer the internal IP address and MAC address of the target mobile device.

**Encrypted Wireless Network** If the target device is associated with an encrypted wireless network, all packet header

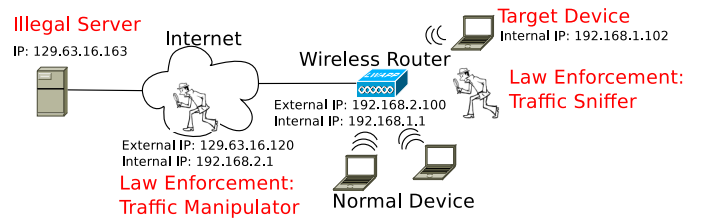


Fig. 1. Basic Idea

information above IP layer (included) is not discernible. In this case, we propose to utilize packet length information from radiotap header in IEEE 802.11 data frames, and develop a novel packet-size based traceback mechanism. That is, law enforcement manipulates size of packets in the suspect public traffic flow. The process resembles embedding a secret signal into the target flow. We expect such signal will not be distorted by NAT or the distortion is predictable. Therefore, if the law enforcement sniffs wireless traffic and recognizes such secret signal, the corresponding wireless frames disclose the MAC address of the suspect mobile. We can also apply this generic technique to traceback under unencrypted wireless network.

### C. Problem Definition

This paper focuses on the traceback in encrypted wireless crime scene investigation.

**Problem Definition:** Assume the law enforcement has identified a public IP address through which a mobile device has established a connection with an illegal website for downloading illegal contents such as child porn. The objective of our traceback technique is to efficiently identify the MAC address of the target mobile device hiding behind encrypted wireless routers.

**Performance Metrics:** As a traceback mechanism designed for the forensics purpose, the performance should be measured in two metrics: (i) *Accuracy*: We apply detection rate and false positive rate to measure the accuracy of our techniques. *Detection rate* is the probability of correctly identifying the target device in a traceback. The *false positive rate* is the probability of falsely identifying an innocent device as the target device. In the paper we present a thorough theoretical and experimental analysis of the correctness of our approach. (ii) *Efficiency*: Since the target device may not remain radioactive for an extended period of time, our traceback mechanism should make decision in an efficient manner. We define efficiency measure as the minimum number of packets required for traceback mechanism to identify a target device.

## III. TRACEBACK SCHEME AGAINST ENCRYPTED WIRELESS COMMUNICATION

It is more difficult to detect target packet flow in an encrypted wireless network. Therefore, we develop a traceback approach based on IEEE 802.11 header and radiotap header in data frames.

### A. Overview of Packet-Based Traceback Mechanism

Figure 2 illustrates the framework of the wireless crime scene investigation. Recall that the basic idea of identifying the criminal traffic and target mobile device is: if the law

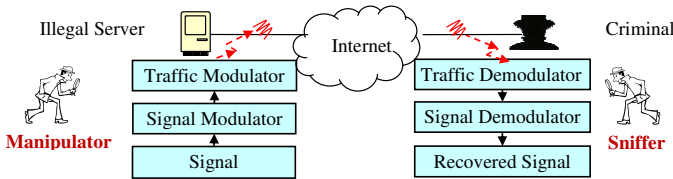


Fig. 2. Traceback Framework

enforcement traffic *manipulator* embeds a signal into the target public traffic on the Internet by manipulating the packet size, then the law enforcement traffic *sniffer* will be able to recover the signal from a private wireless traffic flow, whose packet header discloses the criminal mobile device ID such as the MAC address. We now explain the actions taken by the manipulator and sniffer.

The traffic manipulator picks up a *signal*, a sequence of random bits or a message containing information. Because severe packet loss occurs during sniffing wireless networks, we propose error correcting codes within a *signal modulator* to modulate the original signal. The *traffic modulator* is then responsible for embedding the modulated signal into the target traffic. The question here is how to embed signal bits into the traffic? In our case, we propose to use different packet sizes to represent different bits. For example, packet size 100 for bit ‘0’ and size 400 for bit ‘1’. We may also use different packet sizes to represent different symbols. For example, packet size 100 for symbol ‘00’, and size 200 for symbol ‘01’.

At the traffic sniffer end, the *traffic demodulator* will apply appropriate algorithms to recognize the sequence of packet size by sniffing into the target WLAN associated with the wireless router. Because of packet loss, maybe only a fragment of the packet size sequence can be recovered from the target traffic flow. The *signal demodulator* will then apply error correcting codes to derive the corrected signal from such a fragment. If the corrected signal is equal to the original signal, we say the target wireless flow is confirmed, and the target mobile is identified.

From the description above, we can see that there are three challenging issues to answer: (i) How does the traffic modulator manipulate packets to embed a modulated signal into the target traffic and how does the traffic demodulator recover the packet size sequence? (ii) Which error coding code is good for the signal modulator and demodulator in wireless crime scene investigation to combat packet loss during sniffing in order to guarantee the detection rate?

We will answer these questions qualitatively in the rest of this section and analyze the effectiveness of corresponding strategies in Section IV quantitatively.

## B. Traffic Modulator and Demodulator

1) *Traffic Modulator*: From Figure 2, we know that law enforcement uses different packet sizes to transmit different signal symbols. In our experiments, the symbol is a binary bit 0 or 1. We select two packet sizes for symbol 0 and 1 as follows: We first derive a mass packet size distribution in wireless networks. From this distribution, we choose two packet sizes  $S_0$  and  $S_1$  whose probabilities are relatively low in order to reduce the false positive rate during traffic demodulation.

Traffic Modulator is responsible for embedding a signal into the target traffic. The signal is a sequence of binary bits  $r_1, r_2, \dots, r_m$ . Denote  $P$  as the target packet flow.  $P$  is a sequence of packets, whose sizes are  $ps_1, ps_2, \dots, ps_n, n > m$ . To embed signal bits  $r_1, r_2, \dots, r_m$  into packet flow  $P$ , we randomly choose  $m$  packets from  $P$ . Denote  $C$  as the set of chosen packets,  $\{p_{c_1}, p_{c_2}, \dots, p_{c_m}\}, c_1 < \dots < c_i < \dots < c_m$ , where  $c_i$  is the position of a chosen packet and the corresponding packet size is  $ps_{c_i}$ . The random position is used to prevent continuous manipulation from disrupting TCP packet size dynamics and make the traceback hard to detect.

We now introduce how a signal bit  $r_i$  is embedded to a packet  $p_{c_i}$ , whose size is  $ps_{c_i}$ . Recall we use size  $S_0$  for  $r_i = 0$  and  $S_1$  for  $r_i = 1$ . Denote the required packet size for  $r_i$  as  $S_{r_i}$ . If  $ps_{c_i} \geq S_{r_i}$ , we split  $p_{c_i}$  into two packets,  $p'_{c_i}$  and  $p'_{c_i+1}$ . The size of packet  $p'_{c_j}$  is  $ps'_{c_j} = S_{r_i}$ .  $p'_{c_i+1}$  can be merged with successive packets if necessary to preserve regularity of TCP packet size. If  $ps_{c_j} \leq S_{r_i}$ , we ignore this packet and choose next large enough packet to split and embed  $r_i$  into the target traffic. We also revise packet integrity check and other sensitive information including packet sequence number during packet manipulation. Every time we successfully embed one bit  $r_i$  into one packet  $p_{c_i}$ , we record the timestamp, denoted as  $pt_i$ , and the packet size  $p'_{c_i} = S_{r_i}$ . Denote the list of timestamps as  $P_T = \{pt_1, pt_2, \dots, pt_m\}$ , and the list of packet sizes as  $P_S = \{ps_{c_1}, ps_{c_2}, \dots, ps_{c_m}\}$ . These two lists will be used to traffic demodulation.

2) *Traffic Demodulator*: A traffic demodulator uses a wireless traffic sniffer to collect the target WLAN traffic and identifies the target mobile by recognizing the traffic flow embedded with the secret message. A traffic flow is delimited by traffic source and destination MAC addresses. Therefore, traffic demodulator derives the corresponding MAC address as the suspect mobile’s MAC address.

Details about this process are given as follows. The **first step** is to collect wireless traffic, and categorize packets into different classes by their source and destination MAC addresses,  $T_C = \{class_1, class_2, \dots, class_K\}$ .

Since the watermarked packets are encapsulated into larger frames for wireless transmission, the **second step** of traffic demodulation is to restore every packet size to the original value. For every encrypted data frame, we inspect its encryption header and corresponding beacon frame to identify which encryption algorithm is in use. Then we prune the packet by removing encryption headers and other related values from that frame. Finally, we compute the length of remainder, and add fourteen to it, because an Ethernet header has fourteen bytes. Since the payload of the wireless frame is always encrypted with stream cipher (e.g., AES-counter mode), the length of the payload is not changed. This eases the complication of restoring packet size. At the end of this step, we remove beacon frames and management frames from our traffic collection.

The **third step** has series of tasks described as follows. In each packet class, we check the timestamp of every packet. Considering transmission delay  $\Delta$  and jitter  $\lambda$ , we estimate the

arrival time of the manipulated packets based on the record  $P_T$  and  $P_S$  created by traffic modulator. The timestamp of a received manipulated packet should fall in a range. Denote  $T_{upper_i}$  as  $pt_i + \Delta + \lambda$  and  $T_{lower_i}$  as  $pt_i + \Delta - \lambda$  for the manipulated packet at  $pt_i$  at the traffic modulator. We check all packets whose timestamps fall in these ranges, and try to find packets with expected sizes. If a packet matches timestamp and size requirement, the corresponding signal bit is then recovered. If there is a bit not recovered from the packet class, we denote it by a singular number  $c' \notin \{0, 1\}$ . Thus, we recover one signal (a sequence of message bits) for each packet class. For  $class_i$ , if the hamming distance [11] between the recovered signal and original signal is 0, we return the MAC address as the suspect mobile MAC.

### C. Signal Modulator and Demodulator

In this section, we introduce three error correcting codes, *repetition code*, *Hamming code* and *revised online code*, which recover the embedded signal from a set of received manipulated packets. Because of space limit, we introduce the skeleton of how these error correcting codes are used. Please refer to our technical report [12] for details.

1) *Repetition*: Assume that during wireless traffic sniffing, each packet may be lost independently with probability  $p_{lo}$ . To ensure that we are able to recover message bits  $r_1, r_2, \dots, r_m$  from manipulated packets, we repeat each bit for  $k$  times,  $k > 1$ . The modulated signal after repetition is represented as  $r_{11}, r_{12}, \dots, r_{1k}, r_{21}, r_{22}, \dots, r_{2k}, \dots, r_{m1}, r_{m2}, \dots, r_{mk}$ . We embed this modulated signal into the target packet flow and transmit manipulated packets  $\{p'_{c_{11}}, p'_{c_{12}}, \dots, p'_{c_{1k}}, \dots, p'_{c_{mk}}\}$ . It is obvious that we are able to recover bit  $r_i$  as long as we capture at least one packet from  $\{p'_{c_{i1}}, p'_{c_{i2}}, \dots, p'_{c_{ik}}\}$ .

2) *Hamming Code*: Hamming code [11] is able to correct a single error bit in each codeword. In a codeword, there are  $k$  parity bits and  $2^k - k - 1$  data bits,  $k \geq 3$ .

**Encoding**: First, we randomly generate a sequence of bits as the original signal  $M = \{r_1, r_2, \dots, r_m\}$ . Then we split the message into blocks so that each block  $b_i$  contains  $2^k - k - 1$  bits. We apply Hamming code to each block so that we encode every  $2^k - k - 1$  bits into a  $2^k - 1$  bit codeword. The encoding is accomplished by multiplying each block with a generator matrix  $G$ , i.e.  $c_i = b_i \cdot G$ ,  $1 \leq i \leq \frac{m}{2^k - k - 1}$ . The  $2^0$ th,  $2^1$ th, ...,  $2^{k-1}$ th bits of  $c_i$  are parity bits, others are data bits. Therefore, we get a modulated signal  $M' = \{r'_1, r'_2, \dots, r'_m\}$ ,  $m' = \frac{m(2^k - 1)}{2^k - k - 1}$ , and we embed this modulated signal into the target packet flow.

**Decoding**: Assume the output of traffic demodulator is a signal  $N = \{n_1, n_2, \dots, n_{m'}\}$ ,  $m' = \frac{m(2^k - 1)}{2^k - k - 1}$ . The signal demodulator will process  $N$  and output a demodulated signal  $M''$ . The signal demodulator groups every  $2^k - 1$  bits into a block, e.g. block one  $b'_1$  contains  $n_1, n_2, \dots, n_{2^k - 1}$ . In every block, the signal demodulator will check whether all data bits are captured. Take block  $b'_i$  as an example. If no data bit is lost, then the data bits are extracted and copied into  $M''$ . If any data bit appears to be  $c' \notin \{0, 1\}$ , which implies that bit is lost, signal demodulator replaces  $c'$  by a binary number generated

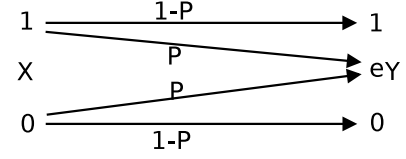


Fig. 3. Channel Model: Binary Erasure Channel

from a pseudo-random number generator, and replaces lost parity bits in the same way. After that, Hamming code is applied to  $b'_i$  for error correcting.

The error correcting step is multiplying a parity check matrix  $H$  and  $b'_i$ , i.e.  $S = H \cdot b'_i$ , where  $S$  is a  $k \times 1$  matrix called the "syndrome". If  $S$  is all zero, then block  $b'_i$  is correct. If  $S$  has a non-zero value, then it points to the position of error bit in  $b'_i$ . Then we flip that bit to correct the error. We extract data bits and copy them to  $M''$ .

This procedure is repeated until every block has been decoded. Finally we compute the hamming distance between  $M$  and  $M''$ . If the distance is 0, the suspect mobile MAC address is identified.

3) *Online Code*: Online code [13] is designed for free erasure channel, in which, there are no constraints on packet loss rate. The underlying channel loses each packet independently with probability  $P$ . Figure 3 represents a model of binary erasure channel [11]. In this model, the input alphabet is  $X$  and output alphabet is  $Y$ . A transmitted 0 (or 1) bit may be lost with probability  $P$ . If we receive a bit 0 (or 1), it is certain that this bit is correct. This is similar to our application scenarios since the wireless router does not change the size of packet. Similarly, during sniffing wireless traffic, we may assume that each packet is lost independently with a probability  $P$ .

A second reason that we apply online code to our approach is because online code is a rateless code, which means there is no constraint on the length of encoded message. Considering that the packet loss probability varies, we adjust certain parameters to meet our demand for high detection rate. The longer the encoding message we generate, the higher is the detection rate we achieve. When we capture a subset of manipulated packets, we are able to recover the original message no matter which subset we get, as long as the size of this subset satisfies decoding requirement.

**Encoding**: Online code encoding scheme has two steps, outer encoding and inner encoding.

*Outer Encoding*: Assume that the original signal is a sequence of binary bits  $r_1, r_2, \dots, r_m$ , where  $m$  is the message length. We first generate a length of  $0.55k\epsilon m$  auxiliary bits  $a_1, a_2, \dots, a_n$ ,  $n = 0.55k\epsilon m$ ,  $k > 1$ . Every auxiliary bit is computed as the exclusive-or (XOR) of a number of original bits. We assign each auxiliary bit  $a_i$  with a degree  $D_i$  and a list of adjacent bit positions  $Ad_i$ . Such information is kept in an outer encoding information table  $T$ . Initially  $D_i$  is 0 and  $Ad_i$  is empty.

Next, we traverse through original bits. For each bit  $r_i$ , we randomly choose  $k$  auxiliary bits  $a'_1, a'_2, \dots, a'_k$ . We then traverse through auxiliary bits and we update  $D_i$ , and  $Ad_i$  for each bit  $a_i$ .

Finally, there may be some auxiliary bits whose degrees

are 0. We delete them and update length  $n$ . We append the auxiliary bits to original bits and they are combined together as composite bits  $c_1, c_2, \dots, c_{n'}, n' = m + n$ .

**Inner Encoding:** The next step is to apply inner encoding to these composite bits. Inner encoding is similar to outer encoding. We generate a piece of check bits  $d_1, d_2, \dots, d_N$ , in which  $d_i$  is computed by XORing a number of  $x$  composite bits. Similarly, we assign degree  $D'_i$  and  $Ad'_i$  to  $d_i$ , which are kept in an inner encoding information table  $T'$ .

We use a pseudo-random number generator  $G(x)$  to generate random number  $x$ . Its probability mass function is given in Equation (1), (2), and (3),

$$F = \lceil \frac{\ln(\epsilon_2/4)}{\ln(1-\epsilon/2)} \rceil, \quad (1)$$

$$p_1 = 1 - \frac{1+1/F}{1+\epsilon}, \quad (2)$$

$$p_i = \frac{(1-p_1)F}{(F-1)i(i-1)}, \quad 2 \leq i \leq F. \quad (3)$$

Finally, we embed this check message  $D$  ( $\{d_1, d_2, \dots, d_N\}$ ) into the target packet flow.

**Decoding:** The inner decoding and outer decoding are quite similar to each other. Here we just show the inner decoding algorithm. Assume that the signal recovered by traffic demodulator is  $D' = \{d'_1, d'_2, \dots, d'_N\}$ . Recall  $d'_i, 1 \leq i \leq N$  is either 0 or 1 if the corresponding manipulated packet is captured.  $d'_i$  is  $c' \notin \{0, 1\}$  if the corresponding manipulated packet is lost. Outer decoding scheme works on bits  $d'_i$  such that  $d'_i \in \{0, 1\}$ . We start decoding those  $d'_j$  with exactly one adjacent composite bit  $m'_k$  that is not decoded yet, and we decode  $m'_k$  by XORing  $d'_j$  and all other decoded bits which are listed in its adjacent table  $Ad'_j$ . This procedure is repeated until we have processed every bit.

After decoding we get a demodulated signal. If the hamming distance between this signal and original signal is 0, we return the MAC addresses.

#### IV. PERFORMANCE ANALYSIS

In this section we analyze detection rate and false positive rate of traceback approaches with different error correcting codes in Section III. Detection rate is the probability that a signal is fully recovered from a target packet flow. False positive rate is the probability that a signal shows up in an unmanipulated traffic flow.

Factors that affect detection rate and false positive rate include the signal length, wireless collision and various fading that cause packet loss during sniffing, and the selection of manipulated packets.

We use the following notations in the theory decryption below. Assume in traffic modulation in Figure 2, we use packet size  $S_0$  for bit 0 and  $S_1$  for bit 1.  $p_{S_0}$  and  $p_{S_1}$  are the two probabilities that the two packet sizes show up in normal traffic respectively.  $m_0$  is the number of 0s and  $m_1$  is the number of 1s in the original signal.  $p_{lo}$  is the packet loss rate. The original signal length is  $m$  bits.

We introduce the theorems for the performance of different error correcting codes. Their proofs can be found in our technical report [12].

##### A. Baseline Approach

In the baseline approach without any error correcting code, based on our decision rule, we need to successfully capture every manipulated packet in a target packet flow. The detection rate is affected by packet loss rate and signal length, as presented in Theorem 1.

**Theorem 1.** *Detection rate  $f_{D_1}(p_{lo}, m)$  of baseline can be calculated as follows,*

$$f_{D_1}(p_{lo}, m) = (1 - p_{lo})^m \quad (4)$$

*The false positive rate  $f_{F_1}(m_0, m_1)$  can be calculated as follows,*

$$f_{F_1}(m_0, m_1) = pr_{S_0}^{m_0} * pr_{S_1}^{m_1} \quad (5)$$

*The redundancy rate of baseline approach is 1.*

We can deduce the following from Theorem 1. (i) Detection rate decreases when packet loss rate or bit length of original signal increases. (ii) When bit length of original signal grows, the false positive rate decreases with increasing signal length. (iii) This baseline approach has no redundancy, but it cannot resist packet loss much.

##### B. Repetition Code

Repetition code repeats every signal several times. Therefore, its detection rate and false positive rate is related with the number of repetitions as is evident in Theorem 2.

**Theorem 2.** *Repetition code repeats every bit  $k$  times. Detection rate  $f_{D_2}(p_{lo}, m, k)$  of repetition code is calculated as follows,*

$$f_{D_2}(p_{lo}, m, k) = (1 - p_{lo}^k)^m \quad (6)$$

*The false positive rate  $f_{F_2}(m_0, m_1, k)$  is calculated in (7).*

$$f_{F_2}(m_0, m_1, k) = (1 - (1 - p_{S_0})^k)^{m_0} * (1 - (1 - p_{S_1})^k)^{m_1} \quad (7)$$

*The redundancy rate  $r_2$  is  $k$ .*

We deduce the following from Theorem 2. (i) If  $k$  increases,  $f_{D_2}(p_{lo}, m, k)$  increases, and the false positive rate  $f_{F_2}(m_0, m_1, k)$  falls. (ii) If  $p_{lo}$  or  $m$  increases,  $f_{D_2}(p_{lo}, m, k)$  decreases. (iii) If  $m_0$  or  $m_1$  increases, the false positive rate falls. (iv) The redundancy is caused by number of repetitions on every bit, so  $r_2$  is  $k$ .

##### C. Hamming Code

Hamming code is able to correct one single error in every codeword, so the detection rate is affected by the length of codeword. It is also affected by packet loss and signal length, as shown in Theorem 3.

**Theorem 3.**  *$k$  is the number of parity bits in every codeword. Detection rate of Hamming code is given in (8),*

$$f_{D_3}(p_{lo}, m, k) = \frac{[(1 - p_{lo})^{2^k - k - 1} + (2^k - k - 1)p_{lo} (1 - p_{lo})^{2^k - 2}]^{\frac{m}{2^k - k - 1}}}{(1 - p_{lo})^{2^k - 2}} \quad (8)$$

Hamming code's redundancy rate is given in (9).

$$r_3 = \frac{2^k - 1}{2^k - k - 1} \quad (9)$$

We deduce the following from Theorem 3. (i) When  $p_{lo}$ ,  $m$  or  $k$  increases, the detection rate decreases. (ii) When  $k$  increases, redundancy rate  $r_3$  decreases. Since it is hard to predict the number of 0s and 1s in a Hamming code encoded message, there is no closed-form formula of false positive rate. In general, a longer message reduces the false positive rate. Our experiments have verified this claim.

#### D. Online Code

Online code includes outer encoding and inner encoding. The parameter  $k$  and  $\epsilon$  affect the modulated signal length and the probability that we recover the original signal from a random fraction of a modulated signal. Theorem 4 gives the performance measurements of online code.

**Theorem 4.**  $n$  is the length of the modulated signal in a target packet flow. Detection rate  $f_{D_4}(\epsilon, k, n, p_{lo})$  of online code is given in (11),

$$\begin{aligned} f_{D_4}(\epsilon, k, n, p_{lo}) &= Pr(n' \geq n_{max})(1 - (\epsilon/2)^{k+1}) \quad (10) \\ &= \sum_{i=0}^{n-n_{max}} \binom{n}{i} p_{lo}^i (1 - p_{lo})^{n-i} * \\ &\quad (1 - (\epsilon/2)^{k+1}) \quad (11) \end{aligned}$$

where  $n_{max}$  is given in (12).

$$n_{max} = \max\left\{ (1 + \epsilon)(1 + 0.55k\epsilon)m, \frac{1 + \epsilon}{\left(\epsilon - \left\lfloor \frac{\ln(1 - \epsilon/2)}{\ln \epsilon^2/4} \right\rfloor\right)(1 - p_{lo})} \right\} \quad (12)$$

the redundancy rate  $r_4$  of online code is given as follows,

$$r_4 \geq n_{max}/m \quad (13)$$

We deduce the following from Theorem 4. (i) If original signal length does not change but modulated signal length increases, the detection rate increases. The redundancy rate increases. (ii) If packet loss rate increases, the detection rate decreases. Since it is hard to predict the number of 0s and 1s in an online code encoded message, there is no closed-form formula of false positive rate. In general, a longer message reduces the false positive rate. Our experiments have verified this claim.

## V. EVALUATION

In this section, we present results from our extensive experiments and simulations to validate our traceback approaches identifying mobiles hiding behind wireless routers. For sniffing, we use a multi-channel wireless sniffer. We will first analyze the performance of this sniffer, including its sniffing capability and system resource usage. Next, we use this device to sniff wireless traffic and derive a packet size mass distribution. Based on this distribution we choose two packet sizes (700 bytes and 1000 bytes), and map bits 0 and 1 to them respectively. This mapping is used in our packet

manipulation scheme. Then, we use extensive simulations and real-world experiments to validate the effectiveness of our error correcting code based traceback.

#### A. Experiment Setup

Figure 1 illustrates the lab environment. The illegal server is an Apache HTTP server. A Lenovo w500 laptop is the target mobile device which downloads a file from the web server. The mobile is connected to a wireless router, and the wireless network is encrypted by Protected Access protocol (WPA). We set up a software router that plays a role of traffic manipulator. The software router is connected to the wireless router at one side and the Internet on the other side. A program runs on the software router which uses the libipq library to embed signals into target traffic.

#### B. Multi-channel Wireless Sniffer

A multi-channel wireless sniffer is critical for wireless network forensics. In general, law enforcement does not have knowledge at which channel a suspect mobile communicates on. Our multi-channel wireless sniffer utilizes 11 AirPcap dongles and are able to capture wireless traffic on 11 802.11 b/g/n channels simultaneously. The sniffing laptop is connected to 3 CACE USB hubs, and each USB hub has 3 or 4 Airpcap dongles. Each dongle can be set to sniff at one channel. The maximum data rate of 802.11 b/g is 54 Mbps. The USB hub supports a speed of 480 Mbps and is able to support 4 Airpcap dongles sniffing simultaneously. The USB interface of the laptop also supports a speed of 480 Mbps. Therefore, this laptop is able to support 11 Airpcap dongles sniffing simultaneously.

Our laptop has enough CPU power and memory to process wireless traffic. Our sniffer contains two i7 processors and each processor has 4 cores. Wireshark is used to sniff wireless traffic. To test if CPU could be saturated, we start two wireshark sessions. One session sniffs via one Airpcap dongle and the other sniffs at the virtual Airpcap interface. The virtual Airpcap interface aggregates traffic from 11 dongles sniffing at 11 different channels respectively. Every wireshark process is executed on a single-core CPU respectively, and we use a VB program to monitor resource usage of these two CPUs. This test lasts for one hour. The result is presented in Figure (4). CPU 1 and CPU 2 are executing one wireshark process independently. It shows that the CPU usage of every wireshark process is quite low. The multi-channel wireless sniffer is capable of processing large data generated during sniffing.

#### C. Packet Loss During Sniffing

Now we demonstrate the packet loss occurring in sniffing. Extensive experiments were been conducted, and observations led to conclusion that packet loss rate changes with time. Many factors contribute towards the time varying packet loss rate. For example, consider a setup where we place our multi-channel wireless sniffer in the lobby. The target device and wireless router are located in an office 10 meters away from the lobby. The packet loss is much more pronounced. The laptop sends ICMP packets to the router so that we can recognize

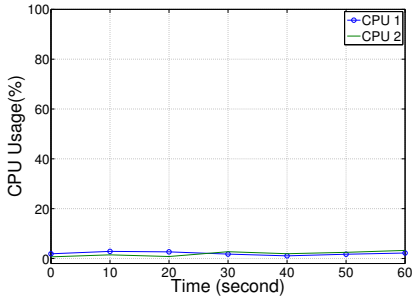


Fig. 4. CPU Usage Test

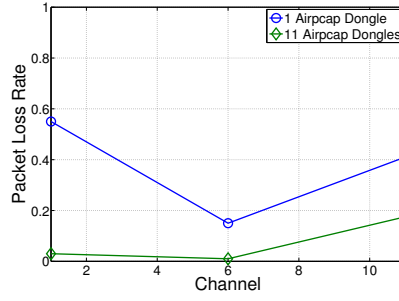


Fig. 5. Packet Loss Rate with 11 Airpcap Dongles Sniffing 1 Channel

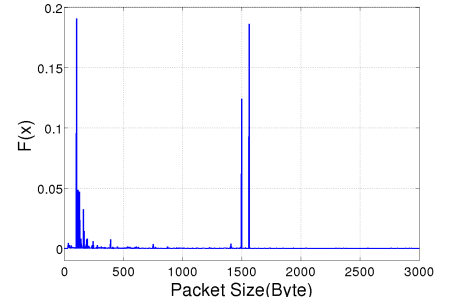


Fig. 6. Campus Wireless Packet Size Probability Mass Function

which packets are lost by checking the ICMP packet counter. For Figure (5), one Wireshark session uses one AirPcap dongle and sniffs at the target channel. A second Wireshark sniffs at the virtual interface with all the 11 dongles sniffing at the target channel. It can be observed that packet loss can be severe in complicated environment. When we use 11 dongles to sniff one channel, the loss rate is reduced. This implies that packet loss is random and 11 dongles increase the chance that a wireless packet is intercepted.

#### D. Selection of Appropriate Packet Size for Message Symbols

The criterion of selecting appropriate packet size for symbols is to reduce the false positive rate. That is, the two packet sizes should appear in reality and their probability should be small. We sniff wireless traffic in all 11 channels simultaneously for one week and save all sniffed packets into pcap files. The packet size mass distribution is given in Figure (6). The probability of packet size of 250 bytes is very high. This is caused by web object tail packets during Internet surfing. The probability of packet size around 1500 bytes is also very high, this is because Ethernet has a Maximum Transmission Unit (MTU) of 1500 bytes or 1492 bytes. Note that packets will be larger when they are encrypted in wireless networks. Figure 7 shows the empirical cumulative distribution function of packet size. Since the packet size density probabilities between 500 bytes and 1000 bytes are very low, we choose packet size of 700 bytes and 1000 bytes for message bits 0 and 1.

#### E. Evaluation by Simulation

Since we cannot control the packet loss rate in reality, in order to compare different traceback strategies, we conducted simulations based on real packet data, which is generated from a long file downloading session.

We conducted two simulations. The first one compares the detection rate of all approaches, and the second one compares the redundancy rate of Hamming code and online code.

In the first simulation, we use a pseudo-random number generator to generate a signal  $M$ . Denote its length to be  $len_M$  bits. We set  $len_M$  to be 10, 11, 12, ..., 40 bits respectively. For every specified value of  $len_M$ , we run the simulation 100000 times. We apply error correcting codes to this signal  $M$  and embed it into target packet flow, or embed it into target packet flow directly. We set packet loss rate  $p_{lo}$  to be 0.30. After each test, we randomly erase  $t$  packets from pcap file,  $t = p_{lo} * N$ ,

$N$  is the total number of packets in this pcap file. Then we try to demodulate the signal from the fraction of pcap file. Let  $S$  to represent the number of success, i.e. the demodulated signal is exactly the original one. We compute detection rate  $R_{detection}$  in this way,  $R_{detection} = S/N_{test}$ , where  $N_{test}$  is the total number of tests.

Figure 8 gives detection rates of all approaches we proposed. The length of signal  $M$  varies from 10 to 40. It can be observed that the baseline approach can not resist packet loss, so its detection rate is zero through all the tests. Hamming(7,4) is better than the baseline approach. The redundancy rate for Hamming(7,4) is about  $7/4 = 1.75$ . Repetition code with repeating time 5 is better than Hamming code. The detection rate of online code is higher than repetition code. We set  $\epsilon$  to be 0.73,  $k$  to be 2, and set the modulated signal length  $n$  to be  $(1 + \epsilon)(1 + 0.55k\epsilon)len_M / (1 - p_{lo})$ . Therefore, the redundancy rate for online code is 4.35. Figure (9) lists the redundancy rates of all different approaches in this simulation.

The second simulation compares the redundancy rates of repetition code and online code. We set the length of signal  $M$  as 25 bits through the whole simulation. Then we increase the packet loss rate from 0.05 to 0.7. For each packet loss rate, we try to maintain a high detection rate (above 0.95). To illustrate our strategy clearly, we take repetition code as an example. When packet loss rate is 0.30, we repeat every bit twice, and run simulation 100000 times. If the average detection rate is above 0.95, we record the redundancy rate as 2, and increase packet loss rate to 0.35 for next test. Otherwise, we keep increasing repetition time by 1 and testing 100000 times until the average detection rate is above 0.95.

Figure 10 shows the result of the second simulation. For online code, we maintain its detection rate to be 0.9514 in average. For repetition code, we maintain its average detection rate to be 0.9523. We can see that when the packet loss rate grows, the redundancy rate of online code grows slower than repetition code. This suggests that we should adopt repetition code when packet loss rate is low, but when packet loss rate is high, online code is more suitable since it will send fewer packets and is able to track shorter sessions.

#### F. Evaluation by Experiments

To demonstrate the effectiveness of our traceback approaches based on error correcting codes in reality, we conducted two groups of experiments with sniffer placed at

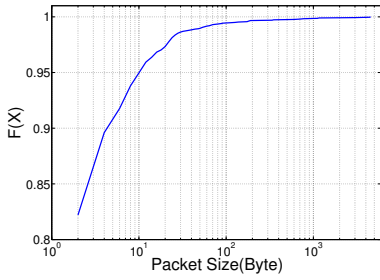


Fig. 7. Packet Size Empirical Cumulative Distribution Function

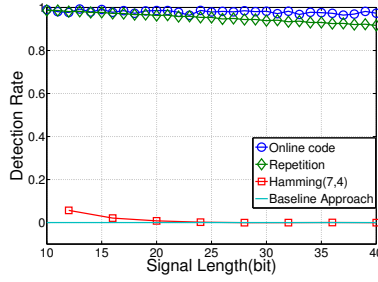


Fig. 8. Simulation: Detection Rate of All Approaches

Error Correcting Code	Redundancy Rate
Baseline Approach	1
Repetition	5
Hamming Code	1.75
Online Code	4.35

Fig. 9. Simulation: Redundancy Rate of All Approaches

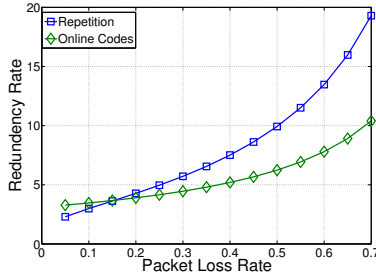


Fig. 10. Redundancy Rate Comparison: Detection Rate of Repetition Code is 0.9523, Detection Rate of Online Code is 0.9514

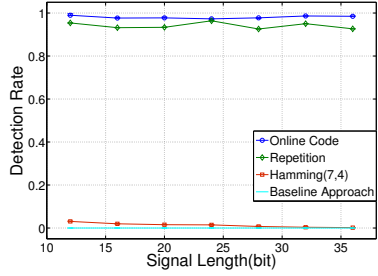


Fig. 11. Experiment: Detection Rate with Sniffer in Neighboring Office

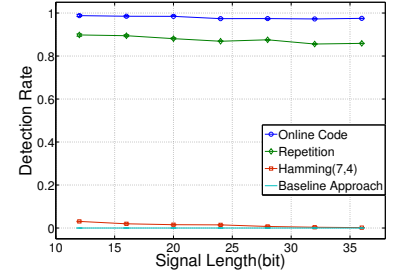


Fig. 12. Experiment: Detection Rate with Sniffer in Lobby

different locations. In each group of tests, we set the signal length to be 12, 16, 20, ..., 36 respectively, and for each specified signal length we repeat the experiments 100 times.

In the first group of experiments, we place the laptop and wireless router in one room, and place sniffer in a neighboring room. Figure 11 gives the detection rate curves of different approaches. The redundancy rate of repetition code is 4, and online code is 4.75. Their detection rates are close to 100%. In the second group of experiments, we place sniffer in the lobby, where the packet loss rate is higher. Figure 12 shows that both repetition code and online code achieve high detection rate while Hamming code and baseline approach have very low detection rate. The redundancy rate of repetition code is 7, and online code is 5.12. Online code has higher detection rate than repetition code. Therefore, online code is the best in this case. The confidence interval is very small and illegible in Figures 11 and 12.

### G. False Positive Rate

We also conducted a set of experiments to test the false positive rate. We place our laptop, wireless router and sniffer in one room. This time the laptop and wireless router are connected through a wired connection. We download the same file and sniff. We try to demodulate signal from intercepted packets with every proposed approach. In our experiments the false positive rates of all approaches are 0 given the long enough message length.

### H. Guideline for Choosing Error Correcting Codes

We derive the following guideline of choosing appropriate error correcting codes in different environments. When packet loss is not severe, Hamming code is suitable because of its low redundancy rate. When packet loss rate becomes high,

repetition code is suitable because its detection rate is high and redundancy rate is lower than online code. When packet loss rate is very high, online code is robust compared to other approaches. Its detection rate is high and redundancy rate is lower than repetition code. These results also validate the correctness of our theorems and analysis in Section IV.

Based on all the simulations and experiments, we conclude that our error correcting codes are effective and necessary. With the multi-channel wireless sniffer, we are able to gain high detection rate with ease.

## VI. RELATED WORK

Because of the space limit, we only review the most related work. Traffic analysis is a common method in correlating traffic around various proxies. Existing traffic analysis can largely be categorized into two groups: passive traffic analysis and active watermarking techniques. Passive traffic analysis techniques have shown how to identify the similarity between server's outbound traffic and client's inbound traffic by recording the traffic passively [14], [15]. The active watermarking techniques intend to embed specific secret signal into the target traffic [16]–[22]. Such techniques can reduce the false positive rate significantly if the signal is long enough and does not require massive training study of traffic cross correlation as required in passive traffic analysis.

In wired networks, traceback across routers have been extensively studied [23]–[27]. There are two types of satieties: revising fields of a packet to create a recognizable mark [23], [25] and logging and hashing packets for later correlation [24], [26], [27]. Particularly, Dean, Franklin and Stubblefield [23] proposed an alternative marking scheme, and used an algebraic approach to reconstruct the packet path traversed over Internet

during a denial of service attack. It is based on error correcting codes and machine learning. Their approach is vulnerable to fake markings by attackers and it also requires large number of packets to reconstruct the attack path.

The most related work to this paper is by Ramsbrock *et al.* [28] where they used packet size based approach to trace through botnet in wired networks. Wireless crime scenes investigated in this paper are different from wired ones. Our work addresses challenges posed by wireless packet loss during sniffing, and multiple WiFi channel monitoring. Our work is also different from other traceback work in wireless networks [29] and [30], [31], which require routers to cooperate and track DDoS attackers in wireless ad-hoc networks.

## VII. CONCLUSION

In this paper, we address the challenge posted by the anonymous WiFi in wireless network forensics, how to identify a suspect mobile hiding behind a wireless router. The problem can be transferred to correlating a private TCP flow in the private NAT wireless network and the corresponding TCP flow on the Internet. We analyzed NAT technique and proposed a variety of novel traceback schemes based on error correcting codes in order to counter packet loss during sniffing. We conducted extensive theoretical analysis and experiments to demonstrate the efficiency and accuracy of our schemes. We also derive a guideline of choosing appropriate error correcting codes in different environments.

## ACKNOWLEDGMENT

This work was supported in part by USA NSF grants 0942113, 0958477, 0943479, and 0907964. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsor agencies.

## REFERENCES

- [1] Z. Liu, Y. Chen, W. Yu, and X. Fu, "Generic network forensic data acquisition from household and small business wireless routers," in *Proceedings of First International Workshop on Data Security and Privacy in wireless Networks (D-SPAN)*, 2010.
- [2] K. Poulsen, "Record 13-year sentence for hacker max vision," <http://www.wired.com/threatlevel/2010/02/max-vision-sentencing/>, 2 2010.
- [3] —, "One hacker's audacious plan to rule the black market in stolen credit cards," [http://www.wired.com/techbiz/people/magazine/17-01/ff\\_max\\_butler?currentPage=2](http://www.wired.com/techbiz/people/magazine/17-01/ff_max_butler?currentPage=2), 12 2008.
- [4] "Aircrack-ng," <http://www.aircrack-ng.org/>, 2010.
- [5] J. Wang, Y. Chen, X. Fu, J. Wang, W. Yu, and N. Zhang, "3DLoc: Three dimensional wireless localization toolkit," in *Proceedings of IEEE ICDCS*, 2010.
- [6] "Geo ip tool - view my ip information," <http://www.geoipool.com/>, 2010.
- [7] The U.S. Department of Justice (DOJ) and National Institute of Justice (NIJ), "Solicitation: Electronic crime and digital evidence recovery," <http://www.ncjrs.gov/pdffiles1/nij/s1000901.pdf>, Apr. 2010.
- [8] S. Rayanchu, A. Mishra, D. Agrawal, S. Saha, and S. Banerjee, "Diagnosing wireless packet losses in 802.11: Separating collision from weak signal," in *Proceedings of IEEE INFOCOM*, 2008.
- [9] S. Guha and P. Francis, "Characterization and measurement of tcp traversal through nats and firewalls," in *Proceedings of Internet Measurement Conference (IMC)*, Oct 2005.
- [10] P. Phaal, "Detecting nat devices using sFlow," <http://www.sflow.org/detectNAT/>, 2010.
- [11] D. J. C. MacKay, "Information theory, inference, and learning algorithms," in *Cambridge University Press*, 2003.
- [12] Y. Chen, Z. Liu, B. Liu, X. Fu, and W. Zhao, "Identifying mobiles hiding behind wireless routers," [http://www.cs.uml.edu/~ychen1/Pkt\\_Sz\\_bsd\\_Traceback.pdf](http://www.cs.uml.edu/~ychen1/Pkt_Sz_bsd_Traceback.pdf), Computer Science Department, University of Massachusetts, Tech. Rep., 2010.
- [13] P. Maymounkov, "online codes," in *New York University Technical Report*, 2002.
- [14] Y. Zhu, X. Fu, B. Graham, R. Bettati, and W. Zhao, "On flow correlation attacks and countermeasures in mix networks," in *Proceedings of Workshop on Privacy Enhancing Technologies (PET)*, May 2004.
- [15] B. N. Levine, M. K. Reiter, C. Wang, and M. Wright, "Timing attacks in low-latency mix-based systems," in *Proceedings of Financial Cryptography (FC)*, February 2004.
- [16] X. Wang and D. S. Reeves, "Robust correlation of encrypted attack traffic through stepping stones by manipulation of inter-packet delays," in *Proceedings of the 2003 ACM Conference on Computer and Communications Security (CCS)*, November 2003.
- [17] X. Wang, S. Chen, and S. Jajodia, "Tracking anonymous peer-to-peer voip calls on the internet," in *Proceedings of the 12th ACM Conference on Computer Communications Security (CCS)*, November 2005.
- [18] P. Peng, P. Ning, and D. S. Reeves, "On the secrecy of timing-based active watermarking trace-back techniques," in *Proceedings of the IEEE Security and Privacy Symposium (S&P)*, May 2006.
- [19] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of tor," in *Proceedings of the IEEE Security and Privacy Symposium (S&P)*, 2006.
- [20] W. Yu, X. Fu, S. Graham, D. Xuan, and W. Zhao, "Dsss-based flow marking technique for invisible traceback," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy (S&P)*, May 2007.
- [21] X. Wang, S. Chen, and S. Jajodia, "Network flow watermarking attack on low-latency anonymous communication systems," in *Proceedings of the IEEE Symposium on Security & Privacy (S&P)*, May 2007.
- [22] Z. Ling, J. Luo, W. Yu, X. Fu, D. Xuan, and W. Jia, "A new cell counter based attack against tor," in *Proceedings of 16th ACM Conference on Computer and Communications Security (CCS)*, November 2009.
- [23] D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to ip traceback," in *Proceedings of Network and Distributed System Security Symposium, NDSS*, February 2001.
- [24] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Hash-based ip traceback," in *Proceedings of ACM SIGCOMM*, 2001.
- [25] D. X. Song and A. Perrig, "Advanced and authenticated marking schemes for ip traceback," in *Proceedings of IEEE INFOCOM*, 2001.
- [26] J. Li, M. Sung, J. Xu, L. Li, and Q. Zhao, "Large-scale ip traceback in high-speed internet: Practical techniques and theoretical foundation," in *Proc. of IEEE Symposium on Security and Privacy*, 2004.
- [27] T. Lee, W. Wu, and W. Huang, "Scalable packet digesting schemes for IP traceback," in *Proc. of IEEE International Conference on Communications (ICC)*, 2004.
- [28] D. Ramsbrock, X. Wang, and X. Jiang, "A first step towards live botmaster traceback," in *Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection (RAID)*, September 2008.
- [29] X. Fu, Y. Zhu, B. Graham, R. Bettati, and W. Zhao, "On flow marking attacks in wireless anonymous communication networks," in *Proceedings of IEEE ICDCS*, 2005.
- [30] M. H. Yang, C.-S. Chiu, and S. Shieh, "Tracing mobile attackers in wireless ad-hoc network," in *Proceedings of Third International Conference on Internet and Web Applications and Services (ICIW)*, 2008.
- [31] V. L. L. Thing and H. C. J. Lee, "Ip traceback for wireless ad-hoc networks," in *Proceedings of the 60th IEEE Vehicular Technology Conference*, 2004.