SECURITY AND COMMUNICATION NETWORKS
Security Comm. Networks 2012; **5**:15–27
Published online 15 February 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/sec.296

SPECIAL ISSUE PAPER

User identification and anonymization in 802.11 wireless LANs

Dingbang Xu[1]*, Yu Wang[2], Xinghua Shi[3] and Xiaohang Yin[1]

[1] Department of Computer Science, Governors State University, University Park, IL, U.S.A.
[2] Department of Computer Science, University of North Carolina at Charlotte, Charlotte, NC, U.S.A.
[3] Harvard Medical School, Harvard University, Boston, MA, U.S.A.

ABSTRACT

Privacy issues have been a serious concern for 802.11 Wireless LAN users. Prior research on privacy issues usually focuses on pseudonym techniques, where the unique, consistent explicit identifiers such as MAC addresses from the users can be frequently changed, and thus it is challenging to track users through those always changing identifiers. However, recent research done by Pang *et al*. (Pang *et al*. Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking, 1997; 99–110) has demonstrated that pseudonyms are not adequate to protect user privacy. The key idea of Pang *et al*.'s method is to locate implicit identifiers (e.g., IP addresses and port numbers a user frequently visits), build user behavior patterns based on these implicit identifiers, and then apply classification techniques to identify users. In this paper, we first propose a new 802.11 user identification approach through enhanced feature selection and generation for building more accurate user behavior patterns. Our simulation results on 9.27 GB SIGCOMM 2004 wireless data sets demonstrate that our method can achieve better classification rates compared with Pang *et al*.'s method. Then, we further study how to provide user anonymity, even if implicit identifiers based identification is applied, by introducing a set of 802.11 user anonymization approaches based on bogus traffic injection. We propose eight different methods to artificially generate bogus data and inject them into original traffic, and thus users' behavior patterns are *disturbed*. Our simulation results on the same SIGCOMM 2004 data sets demonstrate that our anonymization methods can efficiently decrease user identification rates and hence improve user anonymity. Copyright © 2011 John Wiley & Sons, Ltd.

KEYWORDS
privacy; network security; user identification; anonymization; 802.11 networks

***Correspondence**
Dingbang Xu, Department of Computer Science, Governors State University, University Park, IL 60484, U.S.A.
E-mail: d-xu@govst.edu

1. INTRODUCTION

In recent years, various wireless networks have been deployed to support a wide variety of applications. The broadcast nature of wireless networks usually makes it challenging to protect a user's privacy including locations and identities. For example, in a wireless sensor network, the wireless signal strength may partially disclose a user's location, and in an 802.11 wireless network, the MAC address embedded in every frame may partially help identify a user because of the uniqueness of each MAC address. To help protect users' privacy, extensive research on privacy issues has been conducted in different types of wireless networks. For example, Gruteser and Grunwald [1] propose "disposable" MAC addresses to prevent wireless users being continuously tracked. Jiang *et al*. [2] propose obfuscation techniques applying to user identities and transmission time. Along with other techniques such as Refs. [3] and [5], they fall into a broader category of *pseudonym* techniques. The basic idea of these pseudonym techniques is to anonymize the user or device identity through frequently changing pseudonyms. For example, the MAC addresses of wireless devices can be periodically changed to new, unused pseudonyms to prevent users being tracked.

Pseudonym techniques provide the solutions to anonymize *explicit identifiers* such as MAC addresses, however, recent research demonstrates that users' identities may also be disclosed through *implicit identifiers*. To be more specific, Pang *et al*. [4] demonstrate that attackers may identify 64% of all users within 1 day at a spot where 25 users are served simultaneously in every hour. Pang *et al*.'s approach is based on the concept of *implicit identifiers*. The implicit identifiers are those identifiers that may potentially disclose a user's identity. For example, given a group of 100 users accessing wireless networks in a conference with only one user from University of Chicago, if the wireless trace recorded in the conference shows that certain University of Chicago email servers and web servers are frequently visited by some MAC addresses, then the activities from these seemingly unrelated MAC addresses are likely to be correlated to the only user from University of Chicago. Here the IP addresses of those frequently visited email servers and web servers along with their port numbers are implicit identifiers. After identifying certain implicit identifiers, Pang *et al*. further propose to build user behavior patterns based on implicit identifiers in training data sets, and apply data classification techniques to identify users. More details of their method will be reviewed in Section 2.3.

The techniques proposed in this paper can be roughly divided into two parts. In the first part, we focus on user identification in 802.11 wireless LANs (WLANs). We propose a user identification approach through enhanced feature selection and generation, which results in much improved user identification rates compared with the approach by Pang *et al*. [1]. In the second part, considering that privacy issues have been a serious concern in recent years, we go a step beyond user identification, and propose user anonymization methods to protect user privacy in 802.11 WLANs. Assuming that user identification approaches such as our method and Pang *et al*.'s method [1] are employed, we propose multiple ways to generate bogus traffic to "hide" users' behaviors, which may reduce the rates of users being identified. In the following, we give an overview of our user identification and user anonymization approaches.

User identification. Our user identification approach proposed in Section 3 is partially inspired by Pang *et al*.'s method [1]. In particular, we also use the concept of implicit identifiers. However, instead of generating one feature from one implicit identifier in Pang *et al*.'s method, we propose a novel way to dynamically select and generate multiple features from one implicit identifier, and this dynamic feature selection and generation procedure can be controlled through setting up a few parameters. Our approach has three major steps: *feature selection*, *feature generation*, and *classification*. Feature selection decides how many features will be used, and what these features are. Feature generation algorithm processes sample data sets, generates feature values, and then further combines these feature values based on predefined parameters. In the last step, the combined feature values are fed into classification models for training/testing purposes. Since many classification models such as naive Bayes [6] are well defined, we will focus on feature selection and feature generation algorithms in Section 3. We also did an extensive number of simulations using 9.27 GB SIGCOMM 2004 data sets [7] with 49,830,432 wireless packets. Our simulation results in Section 5 show much improved classification rates compared with Pang *et al*.'s [1] method. For example, one simulation result shows an average of 80.6% identification rate for our method, which achieves an 125.1% increase from Pang *et al*.'s method.

User anonymization. In the second part of this paper (Section 4), we aim to study how to improve the user's anonymity even if user identification method based on implicit identifiers is applied. In particular, assuming that Pang *et al*.'s [4] method or our method in Section 3 is employed by the adversary, we investigate new techniques that can decrease user identification rates, and thus improve user anonymity. To achieve this goal, we adopt a natural idea: inject *bogus traffic* into the network, so that users' activities may be "disturbed" and users cannot be identified as they originally are. Carefully designing injection patterns of bogus traffic is critical to achieve user anonymity against 802.11 user identification methods. In Section 4, we propose eight different methods (Methods M_1–M_8) to generate bogus traffic for 802.11 user anonymization, where each of them applies different algorithms and metrics to generate different patterns of bogus packets. To test the effectiveness of these eight methods, we did extensive simulations with the same sets of SIGCOMM 2004 data sets [7]. We evaluate both the effectiveness of each proposed method and how the amount of bogus traffic affects identification rates. We observe that generating the bogus traffic pure-randomly (Method M_1) does not help with anonymization. However, using our proposed advanced methods, the user identification rates can be effectively reduced, even with a small per cent of bogus packets. For example, in one set of simulations with 5% bogus packets, the average user classification rate has decreased from 0.528 to 0.420 (under Method M_7) in Pang *et al*.'s method, and from 0.904 to 0.791 (under Method M_6) in our identification method.

The remainder of this paper is organized as follows. In Section 2, we discuss related work. In Section 3, we present our enhanced user identification method. In Section 4, we present our user anonymization techniques. In Section 5, we provide our simulation results. In Section 6, we conclude this paper and point out some future work.

2. RELATED WORK

Privacy has become a big concern in networking research in recent years. Privacy issues in different types of wireless networks such as wireless LANs [2,3], RFID [8], Bluetooth networks [5], sensor networks [9,10], or general pervasive computing [4] have been extensively studied. The most common solution for preserving location privacy in wireless networks is pseudonym techniques, which are proposed by several researchers such as those in Refs. [2–5]. The basic idea is to anonymize the user or device identity with frequently changed pseudonyms. In this paper, under the assumption that pseudonym techniques are employed, we first propose a novel approach to perfrom user/device identification, then we go a step further to propose user anonymization techniques to protect user privacy. So in this section, we first give an overview of user/device identification methods and anonymization approaches, and then provide an overview of Pang *et al*.'s

[1] method, which inspired our enhanced user identification method.

2.1. User/device identification techniques

Pseudonym techniques solve the problems of identifying users through those explicit identifiers such as MAC addresses. However, recent research [1,11,12] demonstrates that pseudonyms are not adequate to provide anonymity in certain wireless networks. Even without a unique MAC address, attackers may still use different *implicit identifiers* to identify devices or users (wireless fingerprinting techniques). There are two kinds of implicit identifiers: hardware- or software-based. Hardware-based fingerprinting (also called RF fingerprinting) [13,14] uses physical characteristics of RF transmission (such as signal power attenuation, modulation accuracy, or waveform accuracy) to differentiate messages from different radios or different hardware devices. On the other hand, software-based fingerprinting [1,11,12,15] uses the differences in software configurations to distinguish network nodes. For example, Kohno *et al.* [11] estimate the clock skew using TCP and ICMP timestamps to fingerprint network devices. Franklin *et al.* [12] and Desmond *et al.* [15] use statistical analysis of the inter-frame timing of transmitted probe request frames to detect devices. Pang *et al.* [1] study techniques to identify users based on the patterns of the wireless traffic such as network destinations (IP addresses and port numbers) a user frequently visits. In the first part of this paper, we focus on improving Pang *et al.*'s method by proposing new, enhanced feature selection, and generation algorithms.

Some applications may apply machine learning techniques to build implicit identifiers, for example, Internet traffic classification [16–18] and web user identification [19] (based on timing and sizes of web transfer). These approaches are complementary to the ones we mentioned above.

2.2. User/device anonymity techniques

One category of privacy protection techniques in networks is to generate bogus packets/messages artificially and inject them into the real traffic. Our proposed anonymization approach falls into this category. This is not a completely new idea, and has been studied for source location privacy issues in wireless sensor networks [9,20–22], privacy protection in wireless networks through dynamic MAC address exchanging [23], and anonymous services via designing of mix networks [24–26]. We give a few example approaches here.

Kamat *et al.* [9] discuss short-lived fake source routing and persistent fake source routing, where bogus messages are created regularly to mislead adversaries. Shao *et al.* [20] propose Fitted Probabilistic Rate scheme, where dummy messages are created and transmitted in sensor networks, to provide statistically strong source anonymity. Yang *et al.* [21] also propose to apply dummy traffic as well as the additional techniques such as Proxy-based Filtering and Tree-based Filtering to provide source unobservability. A comprehensive survey about privacy-preserving techniques in wireless sensor networks can be found in Ref. [27].

Though the general idea of bogus traffic generation has been used by these approaches as well as our proposed method, how to generate bogus packets (in what patterns) is completely different among these applications, depending largely on the adversaries' threat model in each of these applications. We believe that we are the first one to use this idea and provide detailed methods in 802.11 user anonymization against user pattern-based identification methods such as in Ref. [1].

2.3. 802.11 User identification by Pang *et al.*

Our approach on identifying 802.11 users was partially inspired by the method proposed by Pang *et al.* [1]. Though we use the similar implicit identifiers such as *network destinations*, we use a totally different approach to selecting and generating features before classification models are applied. To help clarify the difference between our approach and Pang *et al.*'s approach [1], we give an overview of their method.

Pang *et al.*'s [1] method assumes that pseudonyms are applied to wireless devices, which means users can change the MAC addresses of their wireless devices at a reasonable time interval (e.g., once an hour). Under this assumption, it is usually not feasible to track/identify users through discovering the correlation among those frequently changed MAC addresses. Instead, to identify different users, Pang *et al.* propose to locate implicit identifiers (e.g., the IP addresses and port numbers a user frequently visits), build user behavior patterns based on these implicit identifiers in the training data sets, and then apply classification techniques to differentiate users.

There are four implicit identifiers proposed in Pang *et al.*'s [1] method: *Network Destination*, *SSID*, *Broadcast Packet Size*, and *MAC Protocol Field*. A network destination is a pair of $\langle$IP, Port$\rangle$ that a user visits. SSID is Service Set Identifier, which is a name broadcast by wireless access points to differentiate wireless LANs. Broadcast packet sizes are the sizes of 802.11 broadcast packets. For example, 276 is the broadcast packet size related to NETBIOS service (UDP port 137). MAC protocol fields represent the fields in MAC frame headers. In 802.11 wireless LANs, there are several protocol fields, for example, *more data*, *order flag*, and *protected flag*.

All these four implicit identifiers are used in classification. More specifically, each implicit identifier will be transformed into one feature, and then assuming that four features are independent, a naive Bayes classifier is applied to identify users. The procedure to generate one feature from one implicit identifier is as follows. Given an implicit identifier $\mathcal{I}$ and a user u, assume that the set of all possible values of $\mathcal{I}$ for u in a training set is S. Then for a sample S_u, the corresponding feature f is computed as

$$f = \frac{\sum_{t\in(S\cap S_u)} \text{weight}(t)}{\sum_{t\in(S\cup S_u)} \text{weight}(t)},$$

where weight $(t) = 1/$(the number of users in S with value t).

3. USER IDENTIFICATION IN 802.11 WLANS: AN APPROACH THROUGH ENHANCED FEATURE SELECTION AND GENERATION

Terms and definitions. Before we present the details of our user identification method, we first informally define a few terms. An *implicit identifier* is a characteristic or a combination of characteristics that may potentially be used to differentiate users. A *feature* is a variable used in the classification model. Given n features $f_1, f_2, \cdots, f_n$, and a special class variable f_c representing the class label, classification, informally speaking, decides the value of f_c based on the values of $f_1, f_2, \cdots, f_n$. Features may represent feature names or feature values depending on the context. Compared with implicit identifiers or features, *attributes* may represent a broader range of concepts. For example, attributes may represent features in classification, or certain field data in network packets, depending on the context.

Our method is partially inspired by Pang *et al.*'s [1] method. We also use those implicit identifiers such as network destinations and SSIDs for 802.11 wireless networks. However, when building user behavior patterns, instead of transforming each implicit identifier into one feature in the classification model, we propose a novel way to dynamically generate multiple features from one implicit identifier, and this dynamic feature generation process can be controlled through setting up a few important parameters. Our simulation result shows improved classification rates compared with the method in Ref. [1].

Roughly speaking, there are two critical algorithms in our method: one is feature selection, and the other is feature generation. Feature selection decides how many features will be used, and what these features are? Based on these selected features, the feature generation algorithm processes sample data sets, generate feature values (i.e., attribute values), and then further group these feature values to reduce the number of feature dimensions. Finally, these grouped feature values will be fed into classification models for training/testing purposes.

To select features for any implicit identifier (e.g., network destination), we first locate the attributes (or attribute combinations) corresponding to the implicit identifier. For example, implicit identifier network destination has a corresponding attribute combination ⟨IP, Port⟩, where these attribute values can be extracted from packet data sets. Next we list all different attribute values for this implicit identifier, and sort them in terms of certain criteria. Lastly, certain number of these attribute values in the sorted list are selected, and each attribute value is transformed into one feature. The details of this feature selection procedure is given in Algorithm 1.

In Algorithm 1, Lines 1–3 are used to select candidate features. Specifically, Line 2 extracts all different attribute values in the data set corresponding to the implicit identifier, and Line 3 further transforms these attribute values into the same number of features. For example, in a data set, for the attribute combination ⟨IP, Port⟩, suppose there are two pairs of different values ⟨123.123.123.123, 80⟩ and ⟨156.156.156.156, 445⟩, we can transform them into two features with the names "*IP* = 123.123.123.123, *Port* = 80" and "*IP* = 156.156.156.156, *Port* = 445," respectively.

Lines 4–9 in Algorithm 1 sort the candidate feature list based on index values. Our design goal for this index value is that it should represent a user's consistent behavior, and also is able to differentiate users. With this design goal, there may exist several different ways to compute index values. As an example, Algorithm 1 shows one approach. Specifically, given any attribute value v'_i, the index value is computed based on the number of the packets having v'_i and the number of users accessing v'_i. Intuitively, if a user's behavior is consistent (e.g., frequently visiting certain web sites), we should be able to observe the corresponding attribute values repeatedly. Additionally, in certain ranges, the lesser the number of users having the same attribute values, the easier the users may be differentiated. This explains why we let index value $\text{Ind}(v'_i) = PC(v'_i)/UC(v'_i)$. Nevertheless, other factors may be incorporated into index value computation, for example, the frequency of attribute values in an unit time duration (e.g., 1 h). After these index values are computed for all candidate features, Line 9 further sorts them based on index values.

In Algorithm 1, the actual feature selection is performed from Lines 10–12. The basic idea here is to select top n features in term of index values. However, we also notice that there may exist some extreme cases, for example, based on the analysis to SIGCOMM 2004 wireless data sets, we observe that most users access the wireless network with SSID *sigcomm-nat*. Though there are huge numbers of packets with SSID *sigcomm-nat*, and the index value for this SSID is also large, this feature does not differentiate users well, because almost every user accesses it. To rule out these extreme cases, we further require that $UC(v_i) < T_{uc}$, where T_{uc} is a pre-set threshold. Notice that the selection of T_{uc} may be critical. If T_{uc} is too high, the selected features may be too common to identify users, but if T_{uc} is too low, the number of selected features may be too small.

Algorithm 1 selects those features that are frequently accessed by users. To identify individual users based on these features, we further apply classification techniques in data mining. However, before classification, we first need to generate feature values for the features selected in Algorithm 1. Algorithm 2 is proposed to generate features (i.e., feature values) based on a user's packet data set. In Algorithm 2, we first generate a bit vector based on the packet data in a unit time period (e.g., 1 h), then reduce the dimensions of bit vectors through combining multiple bits into decimal values.

In Algorithm 2, Steps 1 through 5 are used to compute feature values in a bit vector format. This bit vector is initialized to all 0s in Step 2. Next, each feature is examined based on packet data. If there are some packets having the attribute values corresponding to a feature, then this feature's bit in the bit vector will be set to 1. For example, suppose we have two features "*IP* = 123.123.123.123, *Port* = 80" and "*IP* = 156.156.156.156, *Port* = 445." We examine the packet data set, and find some packets with IP address 123.123.123.123 and port number 80, but there are no packets with IP address 156.156.156.156 and port number 445, then the corresponding bit vector will be set to $\langle 10 \rangle$. In our implementation of this part of Algorithm 2, we build a hash table, which provides mapping between features selected in Algorithm 1 and their indices in the bit vector. This hash table greatly improves the performance of bit setting.

Algorithm 1 Feature Selection for One Implicit Identifier

```
Input: A packet dataset $\mathcal{S}$, an implicit identifier $\mathcal{I}$, a user count threshold $T_{uc}$, and an integer $n$ (i.e., the number of features to be selected).
Output: $n$ features $\mathcal{F}_n^*$ for $\mathcal{I}$.
1: {Candidate Features:}
2: In $\mathcal{S}$, select a complete list $\mathcal{L}'$ of all different values of the attribute (or attributes) corresponding to implicit identifier $\mathcal{I}$. Let $\mathcal{L}' = \{v_1', v_2', \cdots, v_m'\}$ where $m$ is the number of different values of that attribute in $\mathcal{S}$.
3: For each $v_i' \in \mathcal{L}'$, we generate a candidate feature $f_i'$ which is an indicator whether the value of $v_i'$ exists in the dataset for implicit identifier $\mathcal{I}$. Let $\mathcal{F}'$ be the set of all candidate features, i.e., $\mathcal{F}' = \{f_1', f_2', \cdots, f_m'\}$.
4: {Sorting of Features:}
5: for all value $v_i'$ in $\mathcal{L}'$ do
6:     Count $PC(v_i')$, which is the number of packets with value $v_i'$ in $S$.
7:     Count $UC(v_i')$, which is the number of users in $S$ who have packets with value $v_i'$.
8:     Compute index value $Ind(v_i') = \frac{PC(v_i')}{UC(v_i')}$.
9: Decreasingly sort all values $v_i'$ in $\mathcal{L}'$ and their corresponding features $f_i'$ in $\mathcal{F}'$ based on the index values $Ind(v_i')$. Let the ordered lists be $\mathcal{L} = \{v_1, v_2, \cdots, v_m\}$ and $\mathcal{F} = \{f_1, f_2, \cdots, f_m\}$ respectively.
10: {Selection of Features:}
11: Select a list $\mathcal{F}_n^*$ of top $n$ features in the ordered list $\mathcal{F}$, where for each feature $f_i$, its corresponding value $v_i$ in $\mathcal{L}$ need to satisfy $UC(v_i) < T_{uc}$.
12: Return $\mathcal{F}_n^*$.
```

Steps 6 through 10 in Algorithm 2 are used to group the bits in the bit vectors so that the large number of bits can be combined into fewer number of bit groups, and these bit groups are further transformed into decimal numbers. The purpose of these steps is to reduce data dimension, which can facilitate data classification [28]. As an example, let us assume that we have a bit vector with 15 bits $\langle 001010011000011 \rangle$. Further assume that we want to group these 15 bits into three groups, with five bits each. Bit vector $\langle 001010011000011 \rangle$ is grouped into $\langle 00101, 00110, 00011 \rangle$, which will be further transformed into a decimal vector $\langle 5, 6, 3 \rangle$.

After we get these decimal vectors (i.e., feature values) with reduced dimension, we put the data into classification models for either training or testing. The identification of individual users will be performed through classification algorithms, which have been extensively studied with a rich literature.

Algorithm 2 Feature Generation for One Implicit Identifier

```
Input: A packet dataset $\mathcal{S}_u$ from a user $u$ in a unit time period, an implicit identifier $\mathcal{I}$, $n$ features $\mathcal{F}_n^*$ (corresponding to $\mathcal{I}$), and a group number $g$.
Output: A set of feature values $\mathcal{F}_g^* = \{f_1^*, f_2^*, \cdots, f_g^*\}$ .
1: {Calculation of Feature Values:}
2: Initialize a $n$-dimensional vector $X$ with every dimension being 0, i.e., $X = \langle x_1 x_2 \cdots x_n \rangle$ where all $x_i = 0$.
3: for all feature $f_i$ in $\mathcal{F}_n^*$ do
4:     if any packet in $S_u$ has the attribute value $v_i$ which is corresponding to $f_i$ then
5:         Set $x_i$ be 1.
6: {Grouping of Feature Values:}
7: Partition the vector $X$ into $g$ groups, $X_1, X_2, \cdots, X_g$, where each group $X_i$ can be treated as a $\lceil \frac{n}{g} \rceil$-bit binary number. (The last group may have less bits).
8: for $i = 1$ to $g$ do
9:     Transform the binary number $X_i$ into a decimal value $f_i^*$, i.e., $f_i^* = int(X_i)$.
10: Return $\mathcal{F}_g^*$.
```

In Section 5, we present our simulation results on user identification using real-life wireless tracing data, which demostrate that our method has much higher identification rates compared with the one by Pang *et al.* [1].

4. USER ANONYMIZATION IN 802.11 WLANS: AN APPROACH THROUGH INJECTING BOGUS DATA

Since the existing user identification methods based on implicit identifiers may successfully identify unlabeled users, it is also important to investigate user anonymization techniques to protect user privacy in 802.11 networks. In this section, we go a step beyond the proposed user identification methods, and study user anonymization techniques to protect user privacy from these identification methods. Ideally our proposed anonymization techniques will help reduce the user identification rates, even if certain user identification approaches (e.g., our method and Pang *et al.*'s [1] method) are employed.

Before we present the details of our anonymization techniques for 802.11 users, let us first summarize the assumptions here:

(1) Again we assume that pseudonym techniques are applied to wireless devices. For example, MAC addresses of individual devices may be changed every hour.
(2) Adversaries can deploy monitoring devices (eavesdroppers) to observe 802.11 traffic from nearby users. Then existing user identification methods (Ref. [1] or the method in Section 3) based on implicit identifiers, such as *Network Destination* and *SSID*, may be applied, aiming at identifying users.
(3) The traffic between 802.11 users and access points are protected by encryption. Thus, eavesdroppers will not be able to view the payloads of packets/frames. However, they may still derive some information through implicit identifiers from the headers of packets/frames.

The general idea of our anonymization techniques is to generate bogus packets (or frames) and inject them into the original traffic with certain patterns. We assume that each 802.11 device can generate bogus packets which are marked inside payload data. The authorized users can tell what packets are bogus packets when receiving them, and thus ignore them. However, the eavesdroppers cannot tell the difference between real packets and bogus ones by examining the header information.

Assume that the *original traffic* is $\mathcal{S}_o$ with n packets. Our anonymization methods artificially generate the *bogus traffic* $\mathcal{S}_b$ with $p\% \cdot n$ packets, where $p\%$ is an adjustable parameter controlling the amount of bogus packets. The *mixed traffic* $\mathcal{S}_m$ is generated through combining $\mathcal{S}_o$ and $\mathcal{S}_b$. We will study how this mixed traffic ($\mathcal{S}_m$) affects the user identification rate compared with $\mathcal{S}_o$. To be more specific, we will compute the user classification rates applying both identification methods (Ref. [1] and our method in Section 3) on $\mathcal{S}_o$ and $\mathcal{S}_m$, and observe the difference between them if any. A good anonymization technique will significantly reduce the user classification rate by injecting bogus traffic.

The key question of designing our anonymization method is *how to generate the bogus traffic* $\mathcal{S}_b$ *given* $\mathcal{S}_o$. The pattern of bogus traffic will clearly affect the user classification rate of 802.11 user identification methods. Here we focus on the discussion of how to generate bogus attribute values for two specific implicit identifiers: *Network Destination* and *SSID*. These proposed techniques can be easily extended to other attributes or other implicit identifiers. For *Network Destination*, we need to generate artificial pairs of IP addresses and port numbers for bogus packets. For *SSID*, we just need to generate artificial SSIDs for bogus frames. In the following, we propose eight different methods to generate artificial values for implicit identifiers based on different levels of knowledge.

Pure random Method M$_1$. The first method generates attribute values randomly in the whole domain without using any additional knowledge or statistics. Given an attribute domain (i.e., the possible values for this attribute), we randomly generate attribute values with equal probabilities in this domain. For example, for IPv4 addresses in the format of *A.B.C.D* and port numbers, we randomly generate A, B, C, and D from 0 to 255 and port number from 0 to 65 535. We may further restrict the attribute domain if necessary. For example, to generate an Internet-routable address, the address blocks of 10.0.0.0/8, 172.16.0.0/12, or 192.168.0.0/16 should be ruled out from the IP address domain.

Uniformly from global list Method M$_2$. The second method uses a "global" statistical list, which includes all popular attribute values via statistical information in a global or wide-area setting. The attribute values are then generated randomly and uniformly from top k elements in this list. For example, for the implicit identifier *Network Destination*, we first obtain a top $k = 2000$ list of Web sites based on the statistical information (Web traffic ranking) provided by *Alexa* (http://www.alexa.com/). Then we preform nslookup to find the sites' corresponding IP addresses. When generating the bogus packets, we randomly select IP addresses from this list. For simplicity, we set corresponding port numbers to 80 (i.e., only consider http traffic). For the implicit identifier *SSID*, we obtain the statistical information from WiGLE (http://wigle.net/), where top $k = 1000$ SSIDs are provided.

Uniformly from training-set list Method M$_3$. The third method is very simple. The attribute values are uniformly generated from all attribute values appeared in training data sets. Given a training data set, we can first enumerate a list of all different attribute values, then bogus attribute values can be randomly chosen from this list.

Distribution from training-set list Method M$_4$. The fourth method is similar to *Method M$_3$*. The only difference is that, instead of uniformly selecting the attribute values from the training-set list, *Method M$_4$* randomly generates the bogus attribute values following the distribution of attribute values in the training data set. Thus, this method may create more packets with attribute values more frequently observed in the training data set.

The first four methods share the same philosophy, randomly generating the bogus traffic, so that it is hard to distinguish users from each other. In an extreme case, bogus traffic dominates the traffic pattern, thus it is difficult to identify original traffic from mixed traffic. However, that may require large amount of bogus traffic which is not feasible in practice. Therefore, we further refine our anonymization techniques by proposing four more methods, which carefully pick the bogus traffic against the user identification methods (Ref. [1] or our enhanced method).

Uniformly from top packet-count list Method M$_5$. In this method, we try to inject bogus packets with attribute values more frequently observed in the training data set.

Method M_5 is similar to *Method M_3*, except that the training-set list is sorted in decreasing order based on packet count PC(v) from the training data set, and we only take the top k values from the list. Here PC(v) is the number of packets with attribute value v in the training data set. For example, assume that we have three *SSID*s in a training data set: *linksys1*, *linksys2*, and *linksys3*, where the corresponding packet number is 100, 200, and 300, respectively. If we decide to only generate bogus SSIDs from the top $k = 2$ list, then the SSID in each generated frame will be either *linksys3* or *linksys2*.

Uniformly from top packet/user-ratio list Method M_6. The basic procedure in *Method M_6* is similar to that of *Method M_5*. The only difference is that *Method M_6* uses packet/user ratio $R(v)$ as the ranking metric. Here packet/user ratio is defined as $R(v) = \mathrm{PC}(v)/\mathrm{UC}(v)$, where PC($v$) is the number of packets with attribute value v, and UC(v) is the the number of users having the packets with value v in the training data set. To continue our example in *Method M_5*, further assume that the number of users accessing SSIDs *linksys1*, *linksys2*, and *linksys3* is 10, 5, and 40, respectively, then we can get $R(linksys1) = 10$, $R(linksys2) = 40$, and $R(linksys3) = 7.5$. If we still let $k = 2$, then the bogus SSID is randomly selected from $\{linksys2, linksys1\}$.

Uniformly from top reverse-user-count list Method M_7. This method is similar to *Method M_5* and *Method M_6*, but its ranking metric is the inverse of user numbers, which is $1/\mathrm{UC}(v)$. Therefore, the attribute values where less number of users accessing them are on the top of the list, and will be selected as artificial attribute values for bogus packets. This is motivated by an observation that if an attribute value is unique and consistent for a user, then it can be a great indication for identification purpose of that user. In the above example, $1/UC(linksys1) = 0.1$, $1/UC(linksys2) = 0.2$, and $1/UC(linksys3) = 0.025$. Thus, the bogus SSID is uniformly and randomly selected from $\{linksys2, linksys1\}$ if we still let $k = 2$ based on *Method M_7*.

Sequentially from top reverse-user-count list Method M_8. *Method M_8* is a revision to *Method M_7*. In *Method M_8*, we use the same procedure to generate the top k list. However, when generating bogus packets, it picks the values from the list following a sequential order instead of random order.

So far, we only discuss how to artificially create attribute values for those implicit identifiers, since those values will affect user identification rates when applying user identification methods. For other attribute values in packets/frames, they can be generated either randomly or by some existing techniques.

5. SIMULATIONS

5.1. Simulation setup and implementation

To evaluate the effectiveness of our proposed user identification and anonymization methods, we conduct a series of simulations using SIGCOMM 2004 802.11 network data sets [7] (around 9.27 GB wireless packet dump files were processed in the simulations), which were collected through wireless monitors using tcpdump during SIGCOMM 2004 conference. There are 49,830,432 packets being processed via *WireShark* (http://www.wireshark.org/), to extract important attribute values (i.e., packet header fields) such as IP addresses, port numbers, and frame timestamps. We select 10 most active users for training and testing. We summarize the data set being used in Table I. For user identification, we implement both Pang *et al.*'s [1] method and our proposed method which use two implicit identifiers: network destination and SSID. To help us perform data classification, we use WEKA, a data mining software package implemented in JAVA [6] (http://www.cs.waikato.ac.nz/ml/weka/). Similar as in Pang *et al.*'s method, we also assume that pseudonyms are applied to 802.11 devices, which means each user's wireless MAC address can be changed at a reasonable time interval. In our simulation, we assume that pseudonyms are applied once an hour, and we partition each user's wireless trace into multiple subsets with 1-h time duration for each subset. We then process each subset to generate feature values for classification purpose (identifying users).

Table I. Simulation data set summary.

Data source	Wireless traces tcpdumped in SIGCOMM 2004
Data size	9.27 GB
# Packets processed	49,830,432
Implicit identifiers	Network destination and SSID
Training time period	9:00AM-7:00PM 31 August 2004 and 9:00AM-7:00PM 1 September 2004
Testing time period	9:00AM-7:00PM 1 September 2004
# Train/test-ing users	10 most active users

In our simulations, we apply three types of training/testing scenarios for user classification: Type A, Type B, and Type C.

(1) *Type A training/testing*: In this case, we only differentiate two class labels: *Label u* denoting that the instances are from *User u*, and *Label $\bar{u}$* denoting that the instances are *not* from *User u*. During training/testing, Label u instances are from user u, and Label $\bar{u}$ instances are *randomly* selected instances from non-u users.
(2) *Type B training/testing*: Similar to Type A training/testing, Type B training/testing also only differentiates two class labels: *Label u* and *Label $\bar{u}$*. During training/testing, Label u instances are from user u, however, label $\bar{u}$ instances are from *one* non-u user.
(3) Type C training/testing. In this case, we differentiate multiple class labels: *Label u_1* denoting the instances are from *user u_1*, *Label u_2* denoting the instances are from *user u_2*, $\cdots$, and *Label u_k* denoting the instances

are from *user* u_k. During training/testing, label u_i ($1 \leq i \leq k$) instances are from user u_i.

To compare the classification results, we define *classification rate of a user u* as

$$\mathrm{CR}_u = \frac{\#\ \text{Correctly classified instances}}{\#\ \text{Total instances classified}},$$

which is a value between 0 and 1.

As mentioned in Table I, the data set in our simulations spans two days (31 August 2004 and 1 September 2004). Our training uses both days, but only the second day is used for testing. In addition, for all our simulations, we use all available packet data for feature selection, and then select 10 most active users (in terms of wireless activities) for training/testing and compute their average classification rates. For Type B (or Type C) training/testing, we use 10 groups of 2 (or 3) users from the 10 most active users.

5.2. Simulations on user identification

We perform five sets of simulations to evaluate the performance of our method. For all these simulations, average classification rates related to those 10 most active users are plotted in a series of figures (Figures 1–3). All three types of training/testing are executed, and we observe consistent trends in Type B and Type C training/testing. However, the results from Type A training/testing may or may not demonstrate the same trends as those in Type B and Type C training/testing, largely because of the random instance selection performed in Type A training/testing. In the following, we report the results for all these five sets of simulations, mainly focusing on the observations from Type B and Type C training/testing.

Our first set of simulations is to compare the effectiveness of Pang *et al.*'s method and our method. For our method, we set the number of features $n = 100$, the user count threshold $T_{uc} = 36$, and the number of groups $g = 10$. The result is shown in Figure 1. We observe that our method outperforms Pang *et al.*'s method in Type A, Type B, and Type C training/testing for both implicit identifiers (the only exception is Type A training for network destination, where our result

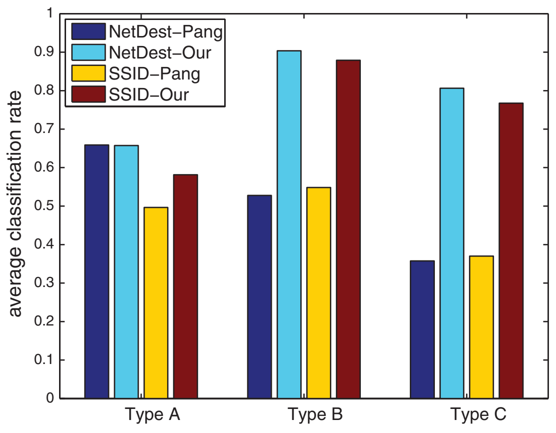

Figure 1. Comparison between Pang *et al.*'s method and our method: average classification rates of user classification.

is similar to the one in Pang *et al.*'s method). Particularly, for network destination implicit identifier in Type C training/testing, our method has the average classification rate of 0.806. Compared with 0.358 from Pang *et al.*'s method, our method increases the average classification rate by 125.1%.

After the first set of simulations confirms that our method is effective in identifying users, we would like to explore our method more deeply. Specifically, we did another four sets of simulations to evaluate our method by setting different parameters.

Our second set of simulations focuses on feature selection (Algorithm 1) with different threshold T_{uc}. We set the number of features $n = 100$, the number of groups $g = 10$, and change the user count thresholds T_{uc} with different values from 10 to 300. The average classification rates for these different T_{uc} values are shown in Figure 2a. We observe that different user count thresholds do affect the classification rates. Setting an appropriate T_{uc} value may increase the rate of user identification. For example, in Type C training/testing for network destination, setting T_{uc} to either 50 or 100 may bring a higher classification rate.

Our third set of simulations is still related to feature selection, but this time we concentrate on the number of features to be selected (i.e., the integer n in Algorithm 1). We set the user count threshold $T_{uc} = 36$, and the number of groups

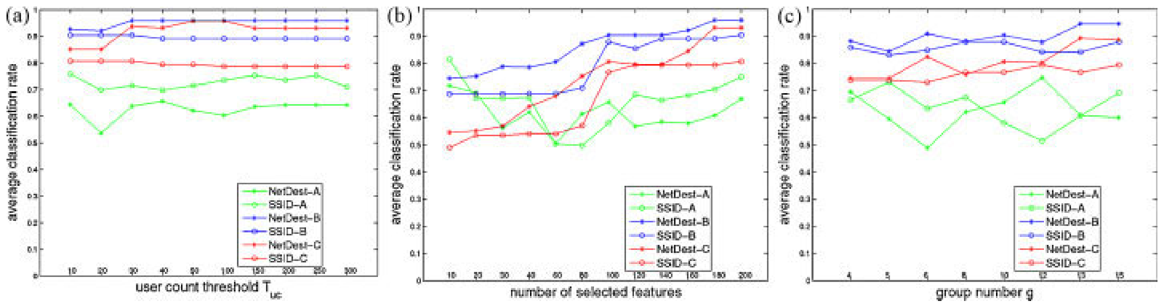

Figure 2. Results of user classification using different threshold T_{uc}, number of features n, or group numbers g. (a) Threshold T_{uc}; (b) number of features n; and (c) group numbers g.

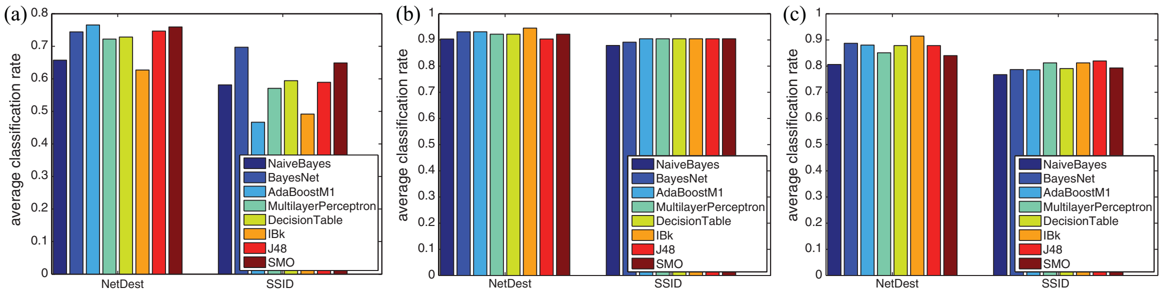

Figure 3. Results of user classification using different classification methods. (a) Type A; (b) Type B; and (c) Type C.

$g = 10$. And then we change the number of features n with different values from 10 to 200. The average classification rates for these different n values are shown in Figure 2b. We observe that different numbers of features can affect classification rates. General speaking, the more features we choose, the better classification results we obtain. This observation is consistent with our intuition. More features selected means that user behaviors can be more precisely (more fine-grained) captured, which usually may bring better results. However, we also notice that Type A training/testing may not necessarily follow this trend. This is because that some instances are randomly selected in Type A training/testing.

Our fourth set of simulations focuses on feature generation. We want to study how group number g (in Algorithm 2) affects classification results. In this set of simulations, we set the number of features $n = 100$, and the user count threshold $T_{uc} = 36$. Then we set g into different values (4,

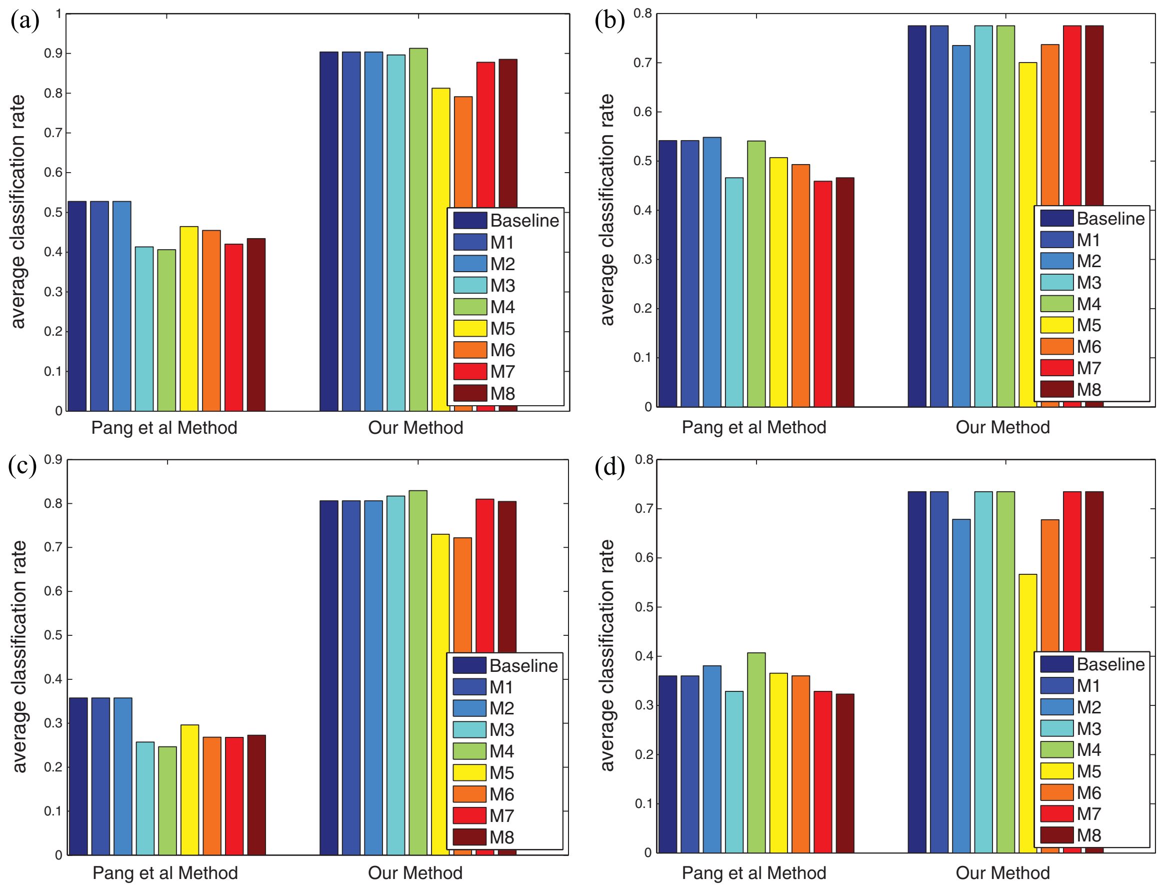

Figure 4. Average classification rates of two identification methods on traffic data generated from eight proposed anonymization methods with 5% bogus packets. (a) NetDest-B; (b) SSID-B; (c) NetDest-C; and (d) SSID-C.

5, 6, 8, 10, 12, 13, and 15). The average classification rates for these different group numbers are shown in Figure 2c. We observe that classification rates are affected by group number g. In certain settings, higher values of group number g may result in higher classification rates. For example, in Type C training/testing for network destination, setting g to 15 has a better average classification rate compared with setting g to 10. However, we also need to notice that this may not always be the case, and it largely may be data set dependent.

Our last set of simulations studies how different data classification methods affect user identification. Similar to the previous simulations, we set the number of features $n = 100$, the user count threshold $T_{uc} = 36$, and the number of groups $g = 10$. Then we select eight classification methods in WEKA: NaiveBayes, BayesNet, AdaBoostM1, MultilayerPerceptron, DecisionTable, IBk, J48, and SMO (the details of these classification methods can be found in Ref. [6]). The average classification rates for these different classification methods are shown in Figure 3. We observe that different classification methods may bring different average classification rates under the same data set. For example, in Type B and Type C training/testing for network destination, the classification method IBk outperforms all the other seven classification methods. But in Type B training/testing for SSID, we do not see big differences among different classification methods. This tells us that classification rates are dependent on classification methods as well as data sets.

5.3. Simulations on user anonymization

To evaluate the effectiveness of our proposed user anonymization techniques, we implement all of them and conduct three sets of simulations. In these simulations, we generate different per cent of bogus packets, and inject them into original traffic. When we preform training, either original or mixed traffic is used. The testing is performed on mixed traffic. In these simulations, only results from Type B and Type C training/testing tasks are reported, since Type A does perform poorly as shown by the previous simulations. Both Pang *et al.*'s method and our enhanced identification method are tested.

In the first set of simulations, we study if injecting a small amount of bogus traffic can decrease the user classification rates, and thus improve the user anonymity. We first train the classifiers using the original traffic data. We then set

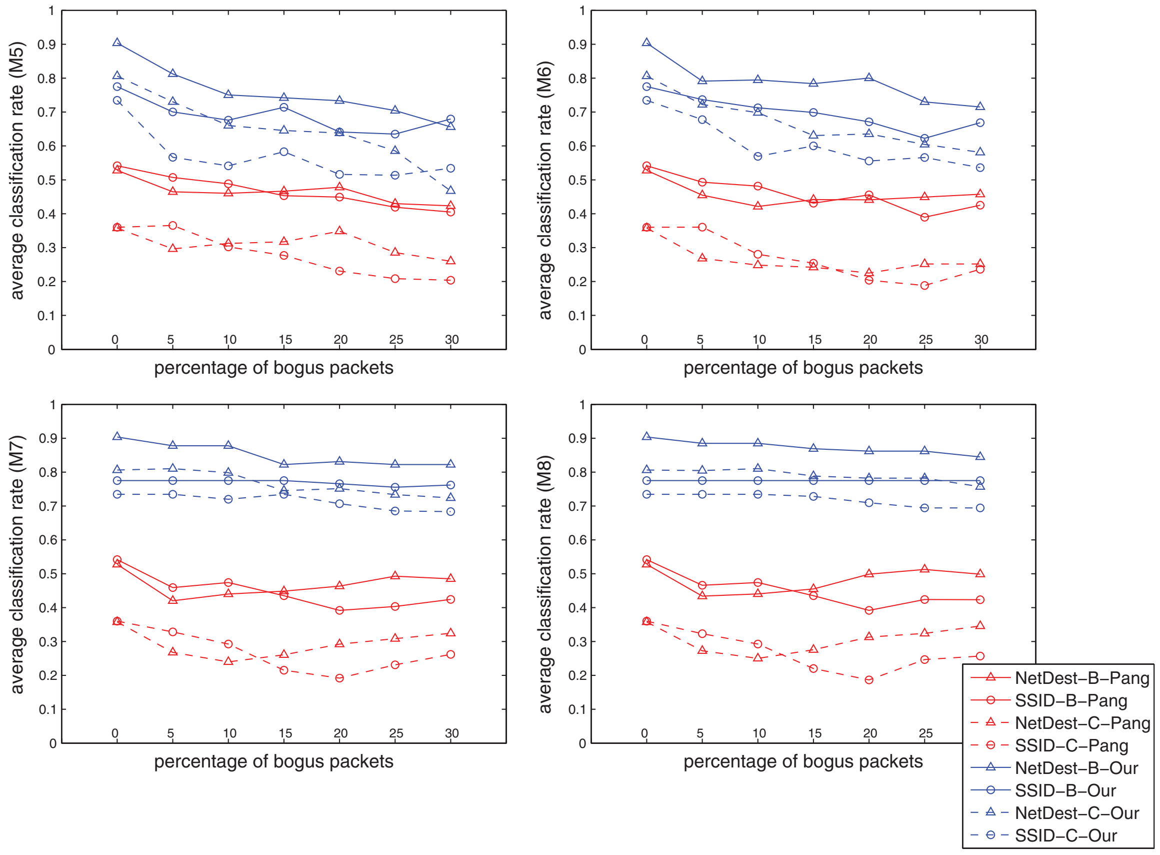

Figure 5. Average classification rates of Pang *et al.*'s method (Pang) and our method (Our) when M_5-M_8 are applied with different amount of bogus packets (0-30% of the original traffic). Here the training sets are original traffic.

the size of the bogus traffic to be 5% of the original traffic, and generate eight bogus traffic sets using proposed methods (*Methods* M_1–M_8). We then perform the testing on the original and eight mixed traffic sets. Figure 4 shows the average classification rates of both identification methods on these traffic sets. For comparison purposes, we mark the testing results from the original traffic sets with *baseline*, and all the other eight testing results are marked with M_1, M_2, ..., M_8, respectively. For example, Figure 4a shows the average classification rates for implicit identifier *Network Destination* on Type B classification task. Hereafter *NetDest-B* denotes implicit identifier *Network Destination* and *Type B* classification task. Based on all four figures in Figure 4, we observe the following:

(1) Our identification method outperforms Pang *et al.*'s [1] approach. Notice that our method uses more features to achieve better classification rates.
(2) Pure random anonymization method (M_1) cannot reduce the classification rates at all. Such pure random bogus traffic can be easily treated as background noise.
(3) Other proposed anonymization techniques can decrease user classification rates in some cases, and hence improve user anonymity. Especially, Methods M_5–M_8 can clearly reduce classification rates in most cases. For example, in Figure 4a, the average classification rate has decreased from 0.528 (baseline) to 0.420 (M_7) in Pang *et al.*'s method, and from 0.904 to 0.791 (M_6) in our method.
(4) Different anonymization methods have different impacts on user anonymity for different user identification methods. For example, we observe that when Pang *et al.*'s method is applied for user identification, *Methods* M_7 and M_8 provide better user anonymity, while when our method is applied, *Methods* M_5 and M_6 provide better anonymity. This is due to different features are used by these user identification methods.

In the second set of simulations, we study the impact of injecting different amount of bogus packets. Based on aforementioned observations, we only focus on *Methods* M_5-M_8. We test the injected amount of bogus packets from 5% to 30% of the original traffic size. The simulation results are shown in Figure 5. In these four figures, we observe that in general, injecting more bogus data results in decreased average classification rates, which means that the user

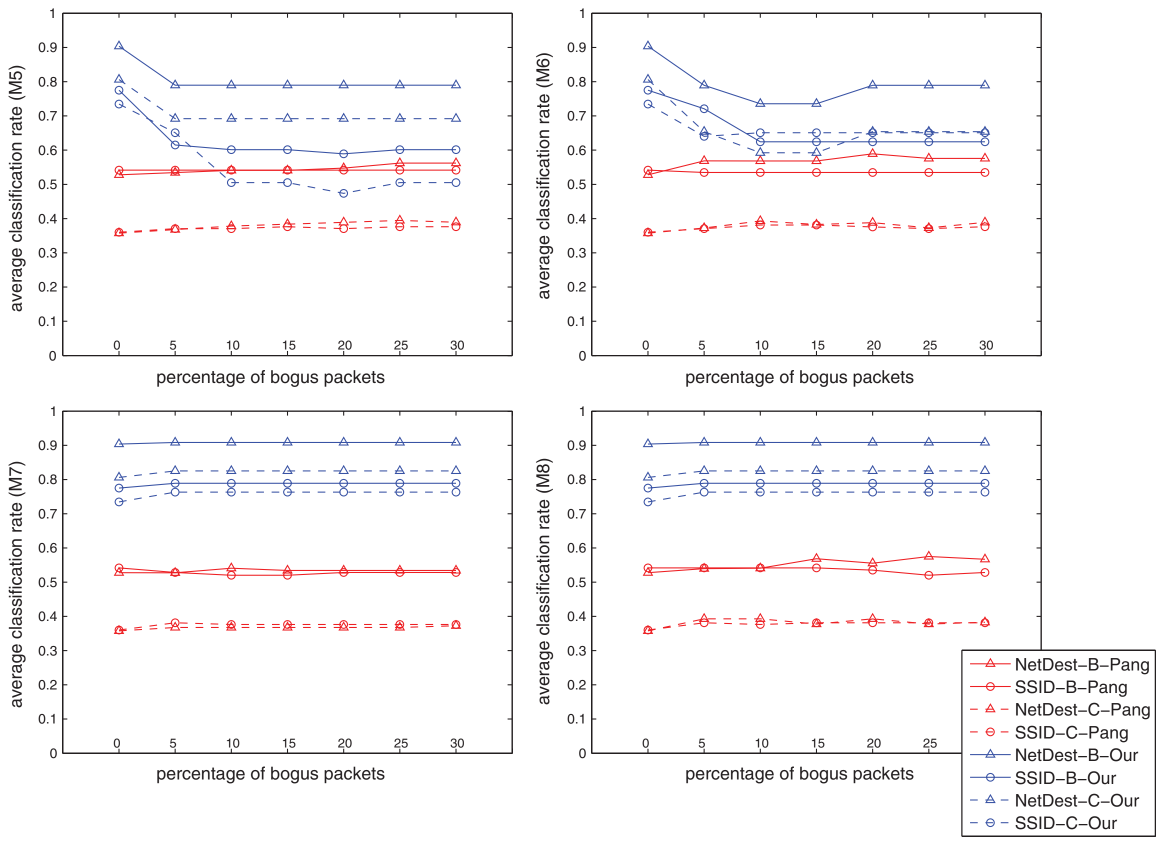

Figure 6. Average classification rates of Pang *et al.*'s method (Pang) and our method (Our) when M_5-M_8 are applied with different amount of bogus packets (0-30% of the original traffic). Here, the training sets are mixed traffic.

anonymity is improved. However, we also observe some exceptions, for example, in the figure related to *Method* M_7, the line marked with *NetDest-B-Pang*. In addition, comparing two user identification methods, the general trend that more bogus packets result in more decreased classification rates is more consistent in our method than that in Pang *et al*.'s [1] method. This may be because the classification rates of Pang *et al*.'s method are almost lower than 50% for Type B and 30% for Type C. Furthermore, even in Pang *et al*.'s [1] method, the results from implicit identifier *SSID* are less irregular than those from *Network Destination*. These observations largely depend on the data sets.

In the third set of simulations, we perform both training and testing using mixed traffic. The simulation results are shown in Figure 6. We have multiple observations from these four figures. (i) When applying *Methods* M_5 and M_6, Our method generally follows the trend that more bogus packets result in more decreased classification rates (note the exceptions exist). (ii) When applying *Methods* M_7 and M_8, injecting more bogus packets almost does not affect the results from our method. This is because in our method, the bogus packets generated from *Methods* M_7 and M_8 may not change the top list of features selected from the data sets. (iii) The lines related to Pang *et al*.'s method are quite flat, which means that there are no significant changes in classification rates. Note that these results may be largely data set dependent.

6. CONCLUSIONS

The techniques proposed in this paper can be divided into two parts: an 802.11 user identification approach through enhanced feature selection and generation, and an 802.11 user anonymization approach through bogus data injection. In the first part, based on the concept of implicit identifiers, we propose a novel approach to selecting and generating features, and thus users can be identified through data classification. Our extensive simulations on 9.27 GB SIGCOMM 2004 wireless trace with 49,830,432 packets demonstrate improved classification rates compared with Pang *et al*.'s [1] method. In the second part, assuming that implicit identifiers based user identification is applied, we focus on the techniques that can decrease user identification rates, and thus improve the user privacy. More specifically, we propose eight different anonymization methods to generate bogus packets, and then inject them into original traffic to decrease the possibility of users being identified. Our simulation results with 9.27 GB SIGCOMM 2004 wireless data sets demonstrate that several of our anonymization methods are effective.

There are several directions we will further explore in our future work. In user identification part, our simulation results preliminarily demonstrate the relationships among different parameters such as *the number of selected features* n, *user count threshold* T_{uc}, and *group number* g. We will look for more data sets for simulations to study the relationships among them. Secondly, since our user identification method is based on the concept of implicit identifiers, we will explore and evaluate other implicit identifiers in our future work. Thirdly, in addition to eight proposed anonymization methods, we will investigate more anonymization methods to help improve user anonymity from more various user identification methods.

ACKNOWLEDGEMENTS

The work of Y. Wang is supported in part by the US National Science Foundation (NSF) under grant no. CNS-0721666, CNS-0915331, and CNS-1050398.

REFERENCES

1. Gruteser M, Grunwald D. Enhancing location privacy in wireless LAN through disposable interface identifiers: a quantitative analysis. *Mobile Network and Applications* 2005; **10**(3): 315–325.
2. Jiang T, Wang HJ, Hu Y-C. Preserving location privacy in wireless LANs. In *MobiSys '07: Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, ACM, New York, NY, U.S.A., 2007; 246–257.
3. Beresford AR, Stajano F. Location Privacy in Pervasive Computing, *IEEE Pervasive Computing* 2003; **2**(1): 46–55.
4. Pang J, Greenstein B, Gummadi R, Seshan S, Wetherall D. 802.11 user fingerprinting. In *MobiCom '07: Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*, ACM, New York, NY, U.S.A., 2007; 99–110.
5. Wong F-L, Stajano F. Location privacy in Bluetooth. In *Proceedings of 2nd European Workshop on Security and Privacy in Ad hoc and Sensor Networks (ESAS '05)*, 2005; 176–188.
6. Witten IH, Frank E. Data Mining: Practical Machine Learning Tools and Techniques (2nd edition), M. Kaufmann, San Francisco, CA, U.S.A., 2005.
7. Rodrig M, Reis C, Mahajan R, Wetherall D, Zahorjan J, Lazowska E. CRAWDAD Data Set uw/sigcomm2004 (v. 2006-10-17), 2006. crawdad.cs.dartmouth.edu/uw/sigcomm2004.
8. Garfinkel SL, Juels A, Pappu R. RFID privacy: an overview of problems and proposed solutions. *IEEE Security and Privacy* 2005; **3**(3): 34–43.
9. Kamat P, Zhang YY, Trappe W, Ozturk C. Enhancing source-location privacy in sensor network routing. In *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS 2005)*, 2005.
10. Mehta K, Liu D, Wright M. Location privacy in sensor networks against a global eavesdropper. In *Proceedings*

of the IEEE International Conference on Network Protocols (ICNP 2007), 2007.

11. Kohno T, Broido A, Claffy KC. Remote physical device fingerprinting. In *SP '05: Proceedings of the 2005 IEEE Symposium on Security and Privacy*, IEEE Computer Society, Washington, DC, U.S.A., 2005; 211–225.
12. Franklin J, McCoy D, Tabriz P, Neagoe V, Van Randwyk J, Sicker D. Passive data link layer 802.11 wireless device driver fingerprinting. In *USENIX-SS'06: Proceedings of the 15th conference on USENIX Security Symposium*, USENIX Association, Berkeley, CA, U.S.A., 2006; 12–12.
13. Faria DB, Cheriton DR. Detecting identity-based attacks in wireless networks using signalprints. In *WiSe '06: Proceedings of the 5th ACM Workshop on Wireless Security*, ACM, New York, NY, U.S.A., 2006; 43–52.
14. Gerdes R, Daniels T, Mina M, Russell S. Device identification via analog signal fingerprinting: a matched filter approach. In *Proceedings of Network & Distributed System Security Symposium (NDSS 2006)*, 2006.
15. Desmond LCC, Yuan CC, Pheng TC, Lee RS. Identifying unique devices through wireless fingerprinting. In *WiSec '08: Proceedings of the First ACM Conference on Wireless Network Security*, ACM, New York, NY, U.S.A., 2008; 46–55.
16. Karagiannis T, Broido A, Faloutsos M, Claffy KC. Transport layer identification of p2p traffic. In *IMC '04: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, ACM, New York, NY, U.S.A., 2004; 121–134.
17. Sun Q, Simon DR, Wang Y-M, Russell W, Padmanabhan VN, Qiu L. Statistical identification of encrypted web browsing traffic. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, IEEE Computer Society, Washington, DC, U.S.A., 2002; 19.
18. Moore AW, Zuev D. Internet traffic classification using bayesian analysis techniques. In *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, ACM, New York, NY, U.S.A., 2005; 50–60.
19. Padmanabhan B, Yang Y. Clickprints on the web: are there signatures in web browsing data? 2006. knowledge.wharton.upenn.edu/papers/1323.pdf.
20. Shao M, Yang Y, Zhu S, Cao G. Towards statistically strong source anonymity for sensor networks. In *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM 2007)*, 2007.
21. Yang Y, Shao M, Zhu S, Urgaonkar B, Cao G. Towards event source unobservability with minimum network traffic in sensor networks. In *WiSec '08: Proceedings of the First ACM Conference on Wireless Network Security*, ACM, New York, NY, U.S.A., 2008; 77–88.
22. Deng J, Han R, Mishra S. Decorrelating wireless sensor network traffic to inhibit traffic analysis attacks. *Pervasive and Mobile Computing* 2006; **2**(2): 159–186.
23. Lei M, Qi ZJ, Hong X, Vrbsky SV. Protecting location privacy with dynamic mac address exchanging in wireless networks. In *Proceedings of IEEE Conference on Intelligence and Security Informatics*, 2007.
24. Shbair WM, Bashandy AR, Shaheen SI. A new security mechanism to perform traffic anonymity with dummy traffic synthesis. In *Proceedings of International Conference on Computational Science and Engineering (CSE 09)*, 2009.
25. Diaz C, Preneel B. Taxonomy of mixes and dummy traffic. In *Proceedings of I-NetSec04: 3rd Working Conference on Privacy and Anonymity in Networked and Distributed Systems*, 2004.
26. Berthold O, Langos H. Dummy traffic against long term intersection attacks. In *Proceedings of Proceedings of Privacy Enhancing Technologies Workshop (PET 2002)*, 2002.
27. Li N, Zhang N, Das SK, Thuraisingham B. Privacy preservation in wireless sensor networks: a state-of-the-art survey. *Ad Hoc Networks* 2009; **7**(8): 1501–1514.
28. Choppin A. Unsupervised classification of high dimensional data by means of self-organizing neural networks. M.Sc. thesis, Universit Catholique de Louvain (Belgium), Computer Science Deptment, 1998.