Journal of Information Processing Vol.23 No.1 81–92 (Jan. 2015)

[DOI: 10.2197/ipsjjip.23.81]

Regular Paper

Unsupervised Clustering-based SPITters Detection Scheme

Kentaroh Toyoda[1,a)] Iwao Sasase[1,b)]

Received: January 30, 2014, Accepted: September 12, 2014

Abstract: VoIP/SIP is taking place of conventional telephony because of very low call charge but it is also attractive for SPITters who advertise or spread phishing calls toward many callees. Although there exist many feature-based SPIT detection methods, none of them provides the flexibility against multiple features and thus complex threshold settings and training phases cannot be avoided. In this paper, we propose an unsupervised and threshold-free SPITters detection scheme based on a clustering algorithm. Our scheme does not use multiple features directly to trap SPITters but uses them to find the dissimilarity among each caller pair and tries to separate the callers into a SPITters cluster and a legitimate one based on the dissimilarity. By computer simulation, we show that the combination of Random Forests dissimilarity and PAM clustering brings the best classification accuracy and our scheme works well when the SPITters account for more than 20% of the entire caller.

Keywords: SPIT, VoIP/SIP security

1. Introduction

Recently, VoIP (Voice over IP) is becoming a major telephony protocol thanks to inexpensive call charge. Unfortunately, this merit for legitimate callers is also beneficial for the caller who merchandises goods or spreads malicious phishing call with recorded voice or an actually human speaking. This type of unsolicited calls is referred to as SPIT (SPam over Internet Telephony) and a SPIT call or SPITters (SPIT callers) detection system should be implemented in a SIP (Session Initiation Protocol) server or an application server on the IMS (IP Multimedia Subsystem).

Many researchers try to find better call behavior features, e.g., call frequency, average call duration, out-degree, and in-degree, to distinguish SPITters from legitimate callers and apply this information to SPITter detection. Although there exist many features, none of the methods employ reasonable solutions to set the threshold and reference models in order to differentiate legitimate callers and SPITters appropriately. In other words, if a new feature is found, it is difficult to integrate the feature into conventional SPIT detection. In addition, it is difficult to obtain training data labeled as “SPITter” or “legitimate caller” in privacy concerns since we have to check the content of calls to confirm whether the caller is legitimate to obtain label. These two shortcomings motivate us to research a flexible SPITters detection approach without threshold-based detection as an unsupervised detection method.

In this paper, we propose an unsupervised and threshold-free SPITters detection scheme by using a clustering algorithm. Our method turns complex threshold setting and training into clustering the callers and identifying the SPITters cluster. We aim to separate the inspected callers into two clusters, one is the legitimate cluster and the other is the SPITters one by using multiple features. Since our scheme leverages the features to find dissimilarity among the callers, any complex threshold settings and training phases can be avoided. Although clustering itself does not give us the SPITter cluster, we identify which cluster is the “SPITters cluster” by comparing the average of a feature e.g., *calls per day*. Since if the callers are clustered well, the callers in one of the cluster call more frequently than the others.

The effectiveness of our scheme depends on a dissimilarity measure and a clustering algorithm. We compare three combination of dissimilarity measures and clustering algorithms, which are (1) *k*-means clustering, (2) Euclidean distance + PAM (Partitioning Around Medoids) clustering, and (3) RF (Random Forests) dissimilarity + PAM clustering. By computer simulation, we show that the combination of RF + PAM is the best choice for SPITter detection since RF effectively finds the importance of features while Euclidean distance does not. We also show that our scheme works well when the SPITters account for more than 20% of entire callers. We also compare our scheme with the well-studied conventional schemes and our RF + PAM scheme outperforms them in terms of true positive rate while maintaining low false positive rate.

The rest of this paper is structured as follows: Section 2 describes the model of SPITters and the system model that we assume in this paper. Related work is summarized in Section 3. The proposed scheme is described in Section 4. Simulation results and evaluation are discussed in Section 5. Finally we conclude our discussion in Section 6.

2. System Model

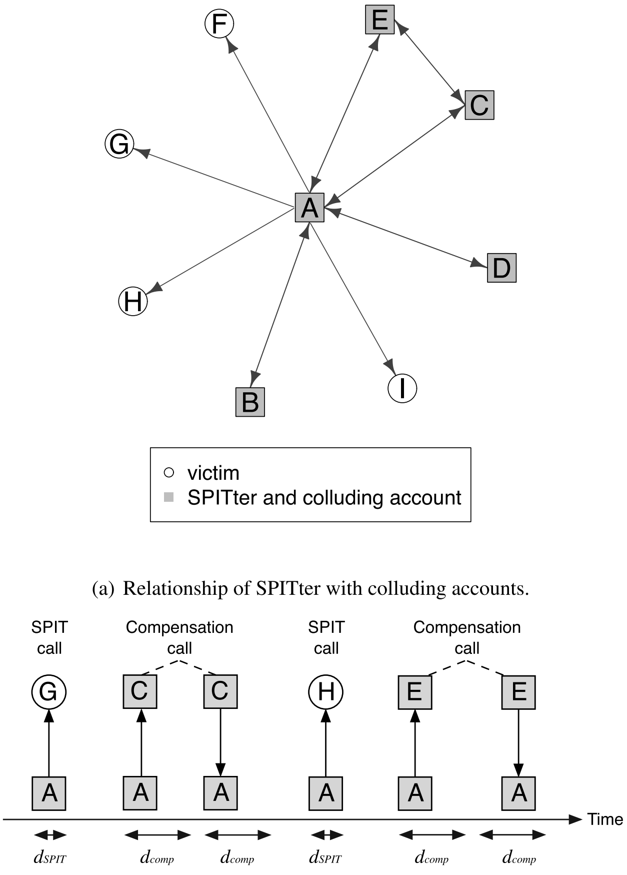
2.1 SPITters Model

SPIT is voice-based spam for advertising merchandise or to commit fraud, such as credit card fraud, to deceive someone to deposit his/her money into their bank accounts. Many researchers define “SPITter” as automatic computer-based SPIT calls genera-

[1] Keio University, Yokohama, Kanagawa 223–8522, Japan

[a)] toyoda@sasase.ics.keio.ac.jp

[b)] sasase@ics.keio.ac.jp



(a) Relationship of SPITter with colluding accounts.

(b) Time series of a SPITter with colluding accounts.

Fig. 1 Graphical model of SPITter with colluding accounts.

tor. The content of SPIT is pre-recorded and automatically played when the call is successfully connected. Most works deal with SPITters who disperse many SPIT calls and calls only victims. However, we have to consider other models of SPITters who own multiple accounts. By using multiple accounts, the SPITters can lower the call frequency, compensate for short average call duration, and make more human-like relationships. **Figures 1** (a) and (b) show an example of the relationship and call behavior of SPITters with colluding accounts, respectively. In Fig. 1 (a), node A is a SPITter and A has four colluding accounts, B, C, D, and E. On the other hand, F, G, H, and I are victim callees. Arrows indicate the direction of calls i.e., SPITter A and the colluding accounts have bi-directional links but the victim callees have uni-directional link from A. From Figs. 1 (a) and (b), we can see that a SPITter with colluding accounts can compensate for short average call duration by occasionally calling back and forth with colluding accounts for a certain duration. Let d_{comp} denote average call duration to compensate for short call duration. We discuss how to set d_{comp} in the following section. By preparing some colluding accounts, a SPITter can imitate the call behavior of a legitimate caller. **Tables 1** (a) and (b) show the call model of SPITters with and without colluding accounts, respectively. In the following section, we discuss how to set the parameters in Tables 1 (a) and (b).

- call frequency (calls per day)
  In order to determine the call frequency model of SPITters, we estimate the maximum amount of SPIT calls per account.

Table 1 Parameters for SPITter model.

(a) without colluding accounts

parameter	value
# of SPIT calls per day	10, 50, 100, 500, and 1,000
SPIT call duration	$d_{\mathrm{SPIT}} \sim \mathit{Exponential}(\mu_{\mathrm{SPIT}} = 15\,\mathrm{sec})$
callee	uniformly chosen
call back rate	0.01

(b) with colluding accounts

parameter	value
# of SPIT calls per day	10, 50, 100, 500, and 1,000
SPIT call duration	$d_{\mathrm{SPIT}} \sim \mathit{Exponential}(\mu_{\mathrm{SPIT}} = 15\,\mathrm{sec})$
callee	uniformly chosen
call back rate	0.01
# of colluding accounts	5
compensation call duration	$d_{comp} \sim \mathit{Exponential}(\mu_{comp})$

  In our assumption, the upper bound of calls per day is estimated under the following situation. We assume that SPITters make calls between 9 to 5. This assumption comes from the following two reasons. The first reason is that SPITters expect the callees to catch as many calls as possible and thus SPIT calls are generated while the most callees are active. The second reason is that most SPIT calls may be generated by a corporation and traditional business hour of corporation is 9 to 5. In reality, human-based SPITter can call as many as 1 or 2 calls per minute: therefore, estimated upper bound per account is 500 or 1,000 calls per day. However, it is too frequent compared with legitimate callers: thus, SPITters are easily detected by a threshold-based SPITter detection [1]. By using multiple accounts, SPITters may avoid a threshold-based SPITter detection. Hence, we also consider low frequency SPITter model that disperses SPIT calls to as many as 10, 50, and 100 calls per day.
- call duration for SPIT call
  For most callees, a received SPIT call must be unsolicited and this could result in a shorter call duration than a legitimate one. Hence we set mean duration of SPIT call μ_{SPIT} to be 15 secs and SPIT call duration d_{SPIT} obeys exponential distribution. This is the same as other related works Refs. [2] and [3].
- callee selection
  Since SPITters aim to broadcast their merchandises or deceive as many victims as possible, we can assume that SPITters seldom call to the same callee again. Thus, we randomly choose the callees.
- call back rate
  We assume that callees intentionally or unintentionally call back to a SPIT caller. The reason for intentional reason is that a callee is simply interested in the content. On the other hand, for unintentional calls, the reason could be that a callee cannot catch the call and calls back later. We consider the above cases and set the call back rate as 0.01.
- # of colluding accounts
  We assume that a SPITter can prepare as many as five colluding accounts. This is because a SPITter can forge Strong Ties (ST) property, which is a ratio of total call duration of the top 5 callees to the total call time.

- call duration for compensation call
  In order to compensate for the short average call duration and the ST property, a sophisticated SPITter can easily forge them by calling with five colluding accounts back and forth. A SPITter with colluding accounts makes not only SPIT calls but also an outgoing call and an incoming call with each colluding account once a day. Thus, if a SPITter prepares five colluding accounts, he/she makes and receives as many as five compensation calls toward/from colluding accounts daily. This can be the case since the calling rate is very low in a VoIP/SIP environment. We set the mean duration for the compensation call as follows. Let d_{target} be the target average call duration by compensation calls and d_{target} is calculated as follows.

$$d_{target} = \frac{\mu_{\mathrm{SPIT}} \times \#(\text{SPIT calls}) + \mu_{comp} \times \#(\text{compensation calls})}{\#(\text{SPIT calls}) + \#(\text{compensation calls})}, \quad (1)$$

where # (X) denotes the number of X. For instance, when the d_{target} is 60 secs and the number of SPIT calls is 100 calls per day, we can calculate μ_{comp} as follows.

$$60 = \frac{15 \times 100 + \mu_{comp} \times 5}{100 + 5},$$
$$\mu_{comp} = 960\,\text{secs}. \quad (2)$$

Note that when the number of SPIT calls is very large e.g., 1,000 calls per day, μ_{comp} is upper bounded since the most active time (9am to 5pm) is spent for SPIT calls. Hence the compensation calls can be exchanged within 16 hours (5pm to 9am) and the upper bound of μ_{comp} is 96 mins (16 hours / (2 × 5)). In this case, d_{target} is calculated from Eq. (1) and $d_{target} \approx 44$ secs. Therefore, we calculate each μ_{comp} subject to an upper bound of d_{target} for a variable number of SPIT calls.

2.2 System Environment

We assume that our SPITters detection system is deployed in a VoIP/SIP service provider and our task is to identify SPITters who belong to its own VoIP/SIP service provider. There exist as many as $N_{callers}$ callers in the VoIP/SIP service provider and thus we can obtain $N_{callers}$ CDR (Call Detail Records) for inspection. We also assume that our SPITters detection scheme is executed at regular intervals (e.g., once a day), and any calls are rejected until the next SPIT detection phase if the caller is judged as a SPITter. This simple method avoids complicated procedures during the call establishment and thus it does not delay the SIP connection.

3. Related Work

There exists much research to combat SPIT and recent works have been summarized [4]. The works are mainly divided into three research categories: (1) features-based SPITters detection scheme e.g., Refs. [1], [3], [5], [6], [7], (2) SPITters detection based on social network trustworthiness e.g., Refs. [8], [9], [10], [11] and (3) content-based SPIT detection e.g., Refs. [12], [13], [14], [15]. However, social network-based SPITter detection and content-based detection have some limitations. In order to calculate the trustworthiness of callers, most social network-based detection schemes construct a graph whose vertices and edges indicate callers and relationships between callers. If all callers (vertices) are within same VoIP/SIP provider, the graph can be constructed from their call history. However, some callers can belong to other providers and thus the graph might be incomplete and it might lead to insufficient trustworthiness. Also, content-based detection schemes cannot avoid privacy concerns. On the other hand, feature-based SPIT detection schemes are not influenced by the above problems, since the features of each caller can be calculated from its own CDR. Therefore, we summarize the feature-based SPITters detection schemes in the following.

Shin et al. suggested Progressive Multi Gray-leveling (PMG) which is a call frequency based SPIT caller detection [1]. They calculate the two gray levels of callers: one for short-term gray level and the other one for long-term gray level. Whether to connect a call is decided by whether the summation of these two levels exceeds its threshold or not. If the summation falls below a certain threshold, the connection is made; otherwise, the connection is blocked. This scheme uses call frequency to distinguish the SPIT callers from legitimate ones. By combining short-term and long-term gray level, a high frequency SPIT caller remains over the threshold and are detected as a SPIT caller. This method needs tuning as many as five parameters.

Yang et al. proposed the supervised decision tree-based SPITters detection [16]. They used six features, which are the number of callees it sends out, ratio of number of calls outgoing and incoming, number of total calls, and number of failed, canceled and completed calls in order to classify the callers. They use labeled training data to construct the decision tree.

Bai et al. pointed out that there is a fundamental difference between legitimate users and spammers on making and receiving calls [5]. A legitimate caller typically makes and receives calls, while a spammer makes a large number of calls but seldom receives calls. Apparently, a small ratio of answered calls and dialed calls can be used to distinguish a legitimate caller and a spammer. Based on the above analysis, they propose three features to identify spam calls, Interaction Ratio (IR), which is the ratio of answered calls to the dialed calls, Historical Ratio (HR), which is the ratio of repeated calls to distinct calls, and Social Ratio (SR), which is the ratio of unknown callees to the total number of callees. They set thresholds X, Y, and Z for IR, HR, and SR, respectively. They compare each threshold with the corresponding features one by one when the caller initiates a call in order to check the legitimacy of a caller.

Bokharaei et al. proposed some features to separate unusual callers from a real phone call dataset in North America [17]. They show that most legitimate callers in their dataset spend most of their talk time with only 4–5 people and they refer to this feature as the ST property. Thus, an ST property is the ratio of the total call duration of the top 5 callees to the total call time. In addition, for most callees, the received SPIT calls must be unsolicited and this could result in shorter call durations than legitimate one. They take advantage of this feature by defining the

Weak Ties (WT) property, which is the fraction of callees that talk for more than 60 secs. The WT value must be very small for SPIT callers since the estimated average SPIT call duration must be shorter than 60 secs. By using these features, they can filter suspicious accounts in their dataset. They introduce F (say 90%) as the threshold against ST and WT and identify the common outstanding callers of ST and WT property as SPITters.

Sengar et al. proposed two SPIT detection methods [3]. In the first approach, they detect high frequency and low call durations callers as SPITters. They prepare the common reference model of legitimate caller whose call arrival $\sim$ *Poisson*(180 secs) and call duration $\sim$ *Exponential*(60 secs). In this approach, they check whether an inspected caller calls five calls within 15 min and if true, they calculate the Mahalanobis distance of the call duration between each inspected caller and the common reference model using the recent n observations. If the distance deviates from the trained threshold, the initiating call is rejected. The second approach focuses on the entropy of the call duration aggregated from the entire call flow. Since most callees soon hang up SPIT calls, the call duration of a SPIT caller is skewed towards a shorter duration and brings about low entropy. Thus, the second approach can detect whether SPIT calls occur in the network.

Wang et al. proposed call/receive ratio and normalized call frequency based features CI and F_{CD} which are input into the k-means clustering algorithm [6]. The scheme finds the center mass of a legitimate callers and classifies each caller by comparing the distance between the caller and a common reference model with the trained threshold.

Although, we notice that there exist many features to distinguish SPITters, none of the methods employs reasonable solutions to set the threshold and to select the reference models in order to differentiate legitimate callers from SPITters. In other words, if we wanted to use a newly found feature or multiple schemes, it would be difficult to integrate the feature into SPIT detection or combine the detection schemes since we have to individually set the thresholds or have to consider a new formula to trap SPITters. Although Amanian et al. proposed to weigh each feature by inferring the effectiveness of the features [18], it still cannot avoid threshold-based detection. Classification-based machine learning approaches can deal with multiple features e.g., decision tree-based detection [16], and may solve the problem. In this case, the training phase is necessary for most classification algorithms. However, it is difficult to obtain training data labeled as “SPITter” or “legitimate caller” due to privacy concerns since we have to check the content of the calls to confirm whether the caller is legitimate to obtain label. The above two shortcomings motivate us to research a SPITters detection approach without threshold-based detection and which is an unsupervised detection method.

4. Proposed Scheme

Here, we propose an unsupervised and threshold-free SPITters detection scheme by using a clustering algorithm. Our method turns complex threshold setting and training problems into clustering the callers and identifying the SPITters cluster which is overall much easier. We try to separate the callers into two clusters, one is legitimate cluster and the other is SPITters based on multiple features. In other words, we do not use the features to directly trap SPITters but to find the dissimilarity among callers. This method avoids the complex threshold tuning and training phase prevalent in conventional works. Although clustering itself does not give us the SPITter cluster, we identify which cluster is a “SPITters cluster” by comparing the average of a feature e.g., *calls per day*, calculated within each cluster. We know the fact that the call duration of SPITters is relatively short compared to legitimate ones and the call frequency of a SPITter is relatively higher than legitimate ones.

The classification accuracy of our scheme highly depends on the combination of dissimilarity measure and clustering algorithm. We introduce three combinations, (1) k-means clustering, (2) Euclidean distance + PAM clustering, and (3) RF dissimilarity and PAM clustering.

Our scheme is superior to conventional schemes in terms of the following points:

- It does not suffer from complicated thresholds tuning.
  As mentioned above, our scheme does not use any thresholds in order to distinguish each inspected caller and thus it is much easier to implement in testbed than traditional schemes.
- It can easily and reasonably adopt new features into a SPITters detection system.
  If a more superior feature were found, most conventional works would tune the threshold of the feature again. In contrast, our scheme can easily involve such a feature since we use the features to find the dissimilarity among callers.
- It neither changes the existing SIP message format nor the SIP terminal at all.
  Our scheme needs only the CDRs of each inspected caller and thus it does not change any SIP format or terminal. It is also an important point for implementation.
- It does not delay SIP connection.
  Our scheme can be executed as an off-line process of the VoIP/SIP service. A SIP server judges whether a caller is legitimate or not by simply checking the classified list.

Our scheme consists of these three procedures and we explain each procedure in detail in the following sections.

(1) calculating features which are considered to differentiate SPITters from legitimate callers;
(2) clustering these callers into two clusters by using the dissimilarity which is calculated from the features; and
(3) deciding which cluster is the “SPITter cluster” by comparing the average of a feature among each cluster.

4.1 Calculating Features

The features represent the characteristic of caller behavior and are calculated from the CDR. **Table 2** shows a simple CDR example of a caller. We calculate as many as $N_{features}$ features for each caller from the CDR for the latest N_{days} days. Let $\boldsymbol{f}$ denote a feature vector of a caller. We occasionally use $\boldsymbol{f}_i$ to indicate caller i’s feature vector. In our setting, we use $N_{features} = 5$ features to describe each caller i.e., $\boldsymbol{f} = (f_{\mathrm{ACD}}, f_{\mathrm{CPD}}, f_{\mathrm{ST}}, f_{\mathrm{WT}}, f_{\mathrm{IOR}})$ since they are considered to be effective in distinguishing SPITters from le-

Table 2 CDR of a caller for $N_{days} = 7$ days.

date[dd/mm/yyyy h:m:s]	caller/callee	direction	duration
01/01/2013 12:02:32	sip:eve@bar.com	outgoing	34”
01/01/2013 13:40:21	sip:dave@foo.com	incoming	45”
···	···	···	···
07/01/2013 21:07:35	sip:dave@foo.com	outgoing	285”

Table 3 Example of feature vectors.

caller	f_{ACD}	f_{CPD}	f_{ST}	f_{WT}	f_{IOR}
Alice	119.65	2.86	0.69	0.61	0.72
Bob	104	6.25	0.66	0.52	0.43
Carrol	61.17	507.12	0.85	0.02	0.1

gitimate callers. Note that if an effective feature was found, it is easily integrated into the feature vector. **Table 3** shows an example of feature vectors. Now we can express each caller’s calling behavior as a feature vector. We cannot say that the five features (f_{ACD}, f_{CPD}, f_{ST}, f_{WT} and f_{IOR}) are sufficient to characterize each caller’s behavior. Certainly, other features can be also considered e.g., missed call ratio, how calls are distributed throughout the day, and the distribution of call hour. However, the aim of our scheme is to show how to use multiple effective features without complex threshold tuning and any training phase. Therefore, we use features that have already been proven to be effective in the literature.

- average call duration (ACD)
  The average call duration is a fundamental feature for the SPITters detection. Since most SPIT calls are unsolicited, this could result in shorter call duration than legitimate ones. We can calculate call duration as follows:

$$f_{\mathrm{ACD}} = \frac{\sum(\mathrm{duration} \mid \mathrm{duration} > 0 \ \&\& \ \mathrm{direction} == \mathrm{outgoing})}{\#(\mathrm{duration} > 0 \ \&\& \ \mathrm{direction} == \mathrm{outgoing})}, \tag{3}$$

  where (X|Y) denotes values X which satisfies conditions Y and #(Y) denotes the number of entries which satisfies conditions Y, respectively.
- call frequency (CPD: calls per day)
  The call frequency is also a fundamental feature for SPITters detection since most SPIT calls make more frequent calls than legitimate callers. We can calculate call frequency as follows:

$$f_{\mathrm{CPD}} = \frac{\#(\mathrm{duration} > 0 \ \&\& \ \mathrm{direction} == \mathrm{outgoing})}{N_{days}}. \tag{4}$$

- ST property (ST)
  The ST property characterizes the fact that most of legitimate callers spend most of their talk time to only 4–5 people. The ST property is the ratio of the total call duration of the top 5 callees to the total call time proposed in Ref. [17].

$$f_{\mathrm{ST}} = \frac{\sum(\mathrm{duration} \mid \mathrm{callee} == \mathrm{top\ 5\ callees})}{\sum(\mathrm{duration})}, \tag{5}$$

  where the “top 5 callees” indicates the five most frequent callees.
- WT property (WT)
  For most callees, the received SPIT calls must be unsolicited and this could result in shorter call durations than legitimate calls. The WT property is the fraction of callees that talk for more than 60 secs. The WT property must be very small for SPIT callers since the estimated average SPIT call duration must be shorter than 60 secs.

$$f_{\mathrm{WT}} = \frac{\#(\mathrm{callee} \mid \overline{\mathrm{duration}} > 60\ \mathrm{secs})}{\#(\mathrm{callee})}, \tag{6}$$

  where $\overline{X}$ denotes the average value of feature X.
- Incoming/Outgoing Ratio (IOR)
  A legitimate caller typically makes and receives calls, while a spammer makes a large number of calls but seldom receives a call. Hence, we can leverage the ratio between incoming calls and outgoing calls as a feature for discrimination. Although many features characterize this fact, e.g., IR, HR, SR [5], CI [6], the ratio between outgoing and incoming calls [16], BDR (Bi-Directional Ratio) and IOR [19], we use IOR in this paper. Both BDR and IOR focus on “the number of callees” instead of “the number of outgoing calls” used in IR, HR, SR, and CI. Because of this, BDR and IOR are more robust against colluding SPITters. In addition, we do not have to use both BDR and IOR since these are very similar properties and we may slightly improve the classification accuracy from the result of Ref. [19] even if we use both. Thus we use only IOR in our scheme.

$$f_{\mathrm{IOR}} = \frac{\#(\mathrm{Incoming})}{\#(\mathrm{Incoming} \cup \mathrm{Outgoing})}, \tag{7}$$

  where Incoming = (callee | direction == incoming) and Outgoing = (callee | direction == outgoing).

After the features are calculated for all callers, we normalize each feature in order to make the features all the same scale. Without normalizing, relatively small feature will be ignored by large value feature. For example, the ST range is [0, 1] but the ACD range is [0, 1,000], as shown in Table 3. Thus the ST is almost ignored because of the difference in magnitude.

4.2 Clustering Callers

Since we aim to separate SPITters from legitimate ones, we cluster the inspected callers into two clusters by using clustering algorithm. Clustering algorithms bundle objects (in our case, callers) who resemble each other. Hence, the callers who have similar feature values resemble each other and are bundled into the same cluster.

Although there exist mainly three major clustering algorithms, hierarchical clustering, *k*-means algorithm, and *k*-medoids algorithm, the candidates we use are *k*-means and *k*-medoids algorithm. This is because most hierarchical clustering algorithms are memory and time exhaustive (the complexities are $O(N_{callers}^3)$ for agglomerative hierarchical clustering or $O(2^{N_{callers}})$ for divisive hierarchical clustering) thus they are not suitable for large datasets. Note that we only have to use one of the following clustering algorithms i.e., *k*-means or *k*-medoids clustering but we introduce both for later comparison.

Algorithm 1 *k*-means clustering.

```
1: Randomly select centroids which are two $N_{features}$-dimensions points and
   indicate the center of cluster 1 and 2, respectively.
2: while No centroids changed do
3:     Assign each caller ($\boldsymbol{f}_i$) to the closer centroid.
4:     Recalculate the two centroids by using newly assigned callers.
5: end while
6: Assign the callers which are closer to centroid 1(2) with cluster 1(2).
```

Algorithm 2 PAM clustering.

```
1: Randomly select two callers as medoids.
2: while No medoids changed do
3:     Assign each caller to the closer medoid.
4:     for each medoid $l \in \{1, 2\}$ do
5:         for each non-medoid vector $m$ do
6:             Swap $l$ and $m$ and calculate the cost based on dissimilarity mea-
               sure.
7:             Select the situation with the lowest cost.
8:         end for
9:     end for
10: end while
11: Assign the callers closer to medoid 1(2) with cluster 1(2).
```

4.2.1 *k*-means Clustering

The simplest clustering method is *k*-means clustering [20]. The algorithm we use is described as Algorithm 1.

The idea is to find two centroids which are the center points of each cluster and to assign each caller to the nearer centroid. The merit of *k*-means is fast and scalable since the calculation time is $O(kN_{callers})$. However, at step 3 in Algorithm 1, Euclidean distance is the only choice for the dissimilarity measure since it has to calculate the distance between callers and center "points" of each cluster. Euclidean distance cannot weigh each feature and thus it does not consider how each feature contribute for clustering.

4.2.2 *k*-medoids Clustering

PAM clustering (Algorithm 2) is the most classical implementation of the *k*-medoids clustering algorithm [21]. In contrast to *k*-means, PAM selects a caller as the center of cluster. This enables us to use variate dissimilarity measures other than Euclidean distance as step 6 in Algorithm 2.

Here, we introduce the other dissimilarity measure, Random Forests (RF) dissimilarity. RF is one of the ensemble decision tree-based classifiers [7]. The reason why we introduce RF dissimilarity is that RF considers the importance of features during tree construction. Although RF is originally a classification algorithm, RF outputs the similarity among callers while constructing decision trees and thus we can leverage the similarity as a dissimilarity measurement. RF constructs as many as T decision trees like **Fig. 2** and uses randomly selected M_{try} out of $N_{features}$ features as the split criterion. The intuition of finding RF similarity is that if the two feature vectors of caller i and caller j are input into the root of a decision tree and both land in the same leaf node, we can see that both are similar to a certain extent and $S_{i,j}$, which is the similarity between caller i and caller j, is increased by one. At the end of the forest construction, the each similarity is output and normalized by the number of trees. Thus we can obtain a similarity-matrix among inspected callers, and each element $S_{i,j}$ takes a [0, 1] value. Then the dissimilarity among caller i and j is obtained as $\sqrt{1 - S_{i,j}}$. Since we can only obtain unlabeled data as shown in Table 3, the problem is how to construct decision trees without labeled data. The recent works Refs. [7] and [22] introduce how to construct RF trees without labeled data. The idea is to label given callers as class 1 and generate "synthetic" callers from the given callers which are labeled as class 2. The synthetic data are used to generate as many as $N_{callers}$ by randomly sampling from the product of the empirical marginal distributions of the features. In Fig. 2, the bar plots underneath the figure indicate the ratio of the class 1 to the class 2 callers who land in the terminal node. In constructing a decision tree, since the feature which is the most distinctive split criterion is selected at each node, the dissimilarity obtained by RF implicitly weighs each feature according to the importance of classification.

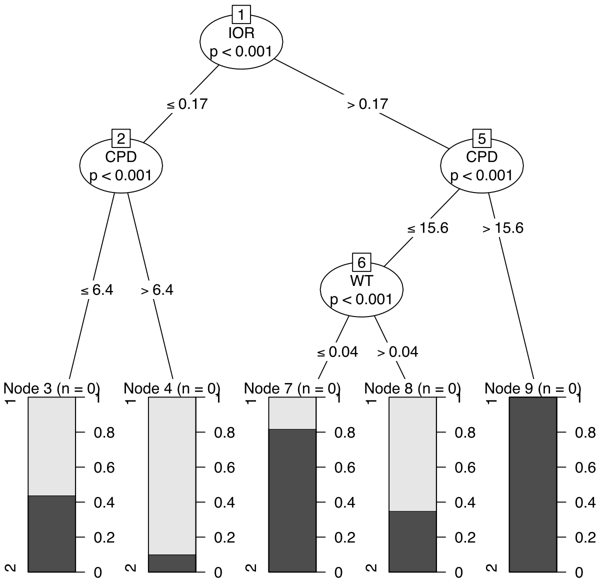


Fig. 2 Example of a decision tree.

Table 4 Outcome of clustering.

caller	cluster
Alice	1
Bob	1
Carrol	2

Table 5 Labeled callers.

caller	cluster
Alice	legitimate caller
Bob	legitimate caller
Carrol	SPITter

As we described above, the merit of PAM is that any dissimilarity can be used for clustering. On the other hand, the complexity of PAM is $O(kN_{callers}^2)$ and is more exhaustive than *k*-means. In order to mitigate the complexity, CLARA (Clustering LARge Application) [21] or CLARANS (CLARA based upon RANdomized Search) [23] can be substituted, which are both the derivative algorithms of PAM.

4.3 Identifying SPITters' Cluster

After clustering the callers, we obtain the callers list labeled as cluster 1 or cluster 2 as shown in **Table 4** but we do not know which is the SPITter cluster. So we have to identify each cluster as shown in **Table 5**. Our scheme accomplishes this by comparing the average value of a feature e.g., f_{CPD} within each cluster. Certainly, we stated that f_{CPD} cannot be simply used to trap every

Table 6 Conditions to identify the SPITter cluster with single feature.

feature	condition
CPD	If the average of CPD in cluster 1 is bigger than that in cluster 2, we can judge cluster 1 as the SPITter cluster.
ACD	If the average of ACD in cluster 1 is less than that in cluster 2, we can judge cluster 1 as the SPITter cluster.
ST	If the average of ST in cluster 1 is less than that in cluster 2, we can judge cluster 1 as the SPITter cluster.
WT	If the average of WT in cluster 1 is less than that in cluster 2, we can judge cluster 1 as the SPITter cluster.
IOR	If the average of IOR in cluster 1 is less than that in cluster 2, we can judge cluster 1 as the SPITter cluster.

type of SPITters but it does not mean that all SPITters call less frequently. In most cases, there exist (traditional) high frequency SPITters: thus a hypothesis that SPITters call more frequently than legitimate callers may still hold. In other words, if the clusters are successfully made, the tendency that SPITters call more frequently than legitimate callers may be observed even though some SPITters are low frequent SPITters. For this reason, we judge the higher f_{CPD} cluster as the SPITter cluster. This way solves the problem of complex threshold tuning, as found in conventional works. Here we introduce f_{CPD} as the representative feature to identify the SPITters cluster. Equation (8) denotes the average f_{CPD} in cluster k.

$$\overline{f_{\mathrm{CPD}}}^{(k)} = \frac{1}{N_k}\sum_{i=1}^{N_k} f^{(k)}_{\mathrm{CPD},i}, \tag{8}$$

where N_k denotes the number of callers in cluster k. For instance, if $\overline{f_{\mathrm{CPD}}}^{(1)}$ is larger than $\overline{f_{\mathrm{CPD}}}^{(2)}$, we identify cluster 1 as the SPITter cluster. Although we use f_{CPD} as the feature to identify the cluster, other features can also do the job. **Table 6** shows each condition of feature to identify the SPITters cluster. In order to confirm the legitimacy of our idea, we compare the classification accuracy by changing the feature to identify the SPITters cluster in Section 5.3.1.

After the above steps, we obtain the callers list whose entries are labeled as "SPITter" or "legitimate caller" as shown in Table 5. Finally, the list is disseminated to its own SIP servers and is used for deciding if the call should be established.

5. Evaluation

We evaluate the classification accuracy against the dataset which consists of real legitimate caller's call logs and self-generated SPIT caller's call data. Classification accuracy has two factors, TP (True Positive rate) and FP (False Positive rate). TP denotes the ratio of correctly identified SPITters and FP rate is the ratio of mistakenly identified legitimate callers as SPITters, respectively. As we mentioned above, there are three combinations of dissimilarity and clustering algorithm, (1) k-means clustering, (2) Euclid. + PAM: Euclidean distance as the dissimilarity with PAM clustering, and (3) RF + PAM: RF dissimilarity with PAM clustering. In the following, we show classification accuracy against N_{days} and compare them against conventional schemes. In addition, we inspect in detail how robust our scheme is when we vary the ratio of SPITters to the entire caller, how correctly each SPITter model is identified, how each feature affects the classification accuracy, and calculation time spent in our method to check scalability.

We assume the following condition: we have the latest N_{days} = 3, 5, 7, 9, or 11 days call logs of the inspected callers and our task is to classify each caller into "SPITter" or "legitimate caller" in an off-line fashion: thus, we do not simulate actual VoIP/SIP messages. We use *R* version 3.0.2, the randomForest package [24] to execute the Random Forests classifier and the PAM clustering package [25] to implement our scheme. All simulations are executed on an off-the-shelf computer which has 3.4 GHz quad cores CPU and 16 GB RAM. Each result is obtained by repeated simulation with as many as 100 trials.

5.1 Dataset

We use the Reality Mining dataset as the legitimate caller's call data, which includes 94 callers' call data as collected by the MIT Media Lab [26]. In the dataset, 68 were colleagues working in the building on MIT campus (90% graduate students, 10% staff) while the remaining 26 callers were incoming students at the university's business school. Although there exist two other call log datasets [27], [28], MIT Media Lab's Reality Mining dataset is the best choice for evaluation. In Ref. [27], the datasets are based on anonymized Call Detail Records (CDR) of phone calls and SMS exchanges between five million of Orange's customers in Ivory Coast between December 1, 2011 and April 28, 2012. However the dataset is available for the NetMob [*1] conference and cannot be used for other purposes. The other one, Nodobo [28], consists of smartphone usage logs of 27 students in a Scottish state high school but 27 persons' data are too small to evaluate the performances of our schemes. Although the Reality Mining dataset is not VoIP call logs but on mobile phone call logs, we assume that VoIP takes place in a conventional telephony network and thus call characteristics of VoIP call is the same as that of mobile phone telephony. On the other hand, to the best of our knowledge, no SPITter's dataset is publicly disclosed. Hence we artificially generate SPITters call logs from the model described in Section 2 and then we mix them with the legitimate callers dataset for evaluation.

We randomly choose 100 callers (including both legitimate callers and SPITters) in the simulation except for Section 5.4. Let $R_{SPITters}$ denote the ratio of SPITters to all callers. We set $R_{SPITters}$ = 0.2. Therefore, we randomly select 80 legitimate callers and 20 SPITters. It is a similar setting to Ref. [11] which considers 25% of all callers as SPITters. It seems to excessive to consider the cost to obtain SIP addresses. However, nowadays, many SIP service providers emerge that offer free SIP accounts e.g., Call Centric [*2], Voiptalk [*3] and Onsip [*4]. In addition, as the cost of obtaining E-mail accounts has dropped in the last decade, the cost of obtaining VoIP/SIP accounts might drop due to price competition in the near future. Therefore, there can be the case that SPITters account for 20% of all callers. However, since no VoIP/SIP calls/callers statistics are publicly available, we cannot guess how many SPITters account for all callers and thus we can-

[*1] http://www.netmob.org/

[*2] http://www.callcentric.com/

[*3] https://www.voiptalk.org

[*4] http://www.onsip.com/about-voip/sip/sip-account

not conclude whether the setting we choose i.e, $R_{SPITters} = 0.2$, is reasonable. Therefore, we vary $R_{SPITters}$ from 1% to 50% and evaluate the classification accuracy in Section 5.3.3.

5.2 Parameter Tuning for RF

Before our simulations, we tune the two parameters M_{try} and T to find the RF dissimilarity and use them for the following evaluation. In this setting, we use 7 days' worth of call logs.

We first tune M_{try} under the situation of $T = 25$ which is the recommended choice for tuning T [7]. **Figure 3** (a) shows classification accuracy versus M_{try}. Since we use five features, M_{try} can take on the value 1 through 5. From Fig. 3 (a), we can see that both TP and FP get slightly better as M_{try} gets larger. In our situation, the number of features is relatively small and thus we use $M_{try} = 5$ in the following simulation.

We also tune T, which is the number of generated trees. Figure 3 (b) shows the classification accuracy versus T. From Fig. 3 (b), we can see that both TP and FP get better as T get larger and that we achieve TP = 0.95 and FP = 0.014 when $T = 500$ but both TP and FP are consistent when $T \geq 500$. Figure 3 (c) shows the calculation time to find dissimilarities among callers versus log-scaled T. From Fig. 3 (c), we can see that the calculation time linearly increases when T gets larger. From these results, we do not have to use $T > 500$ by considering the balance between calculation time and accuracy: therefore, we use $T = 500$ in the following simulation.

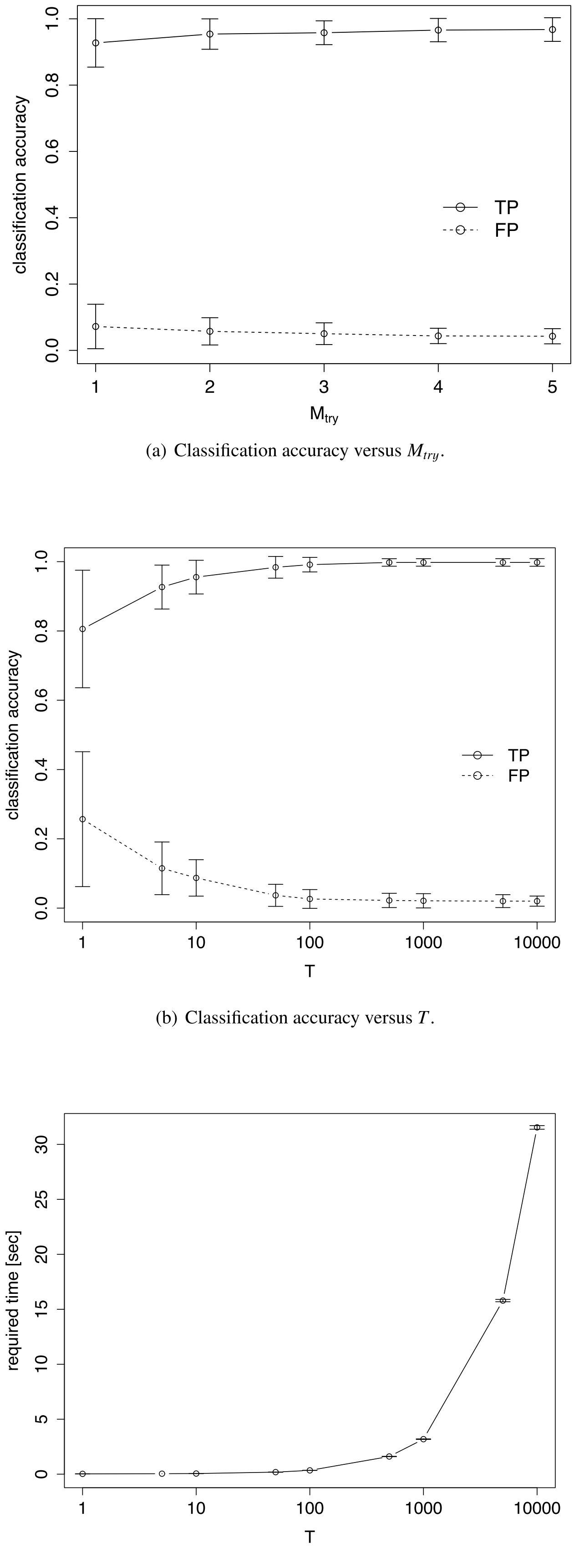
5.3 Classification Accuracy

5.3.1 Classification Accuracy versus Chosen Feature to Identify the SPITter Cluster

In Section 4.3, we stated that any feature can be used to identify the SPITter cluster after clustering. In order to clarify this notion, we compare the classification accuracy versus the feature used to identify the SPITter cluster in **Fig. 4**. We use the RF + PAM scheme where $T = 500$ and $M_{try} = 5$, $N_{callers} = 100$, and $R_{SPITters} = 0.2$ and follow the condition of each feature to identify the SPITter cluster in Table 6. From Fig. 4, our scheme can achieve nearly the same TP and FP regardless of the chosen feature and thus any feature can be used to identify the SPITter cluster. From this result, we can say that the SPITter cluster can be identified by comparing the average of any single feature if the clustering is successfully done.

5.3.2 Classification Accuracy versus N_{days}

We compare our methods with LTD [17] and PMG [1]. The reason why we chose LTD and PMG for comparison is because both methods are well-studied works and the authors gave clear descriptions for deciding the threshold setting and algorithm. Although there exist many other feature-based SPIT detection methods e.g., Refs. [3], [5], [6], and [16], they give ambiguous setting for the threshold, reference model, or training method: thus, we do not compare them in order to avoid inaccurate comparisons. Both LTD and PMG need threshold tuning. Although LTD needs one parameter F and $F = 0.9$ is suggested in their original work, we use $F = 0.7$ which give more accurate detection against our dataset. For PMG, five parameters $TL1$, $TL2$, $C1$, $C2$, and T need to be tuned. Two settings for the parameters are suggested in their work [1]. Setting 1 is $(TL1, TL2, C1, C2, T) = (1\,\text{min}, 1\,\text{hour}, 3, 1, 1{,}000)$ and setting 2



(a) Classification accuracy versus M_{try}.

(b) Classification accuracy versus T.

(c) Required time in RF versus T.

Fig. 3 Parameters tuning for RF + PAM.

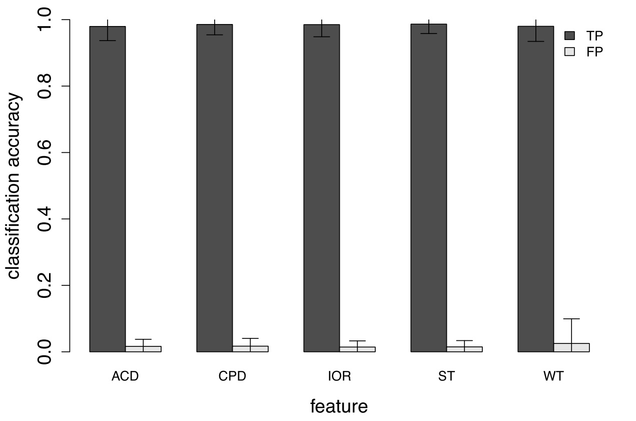
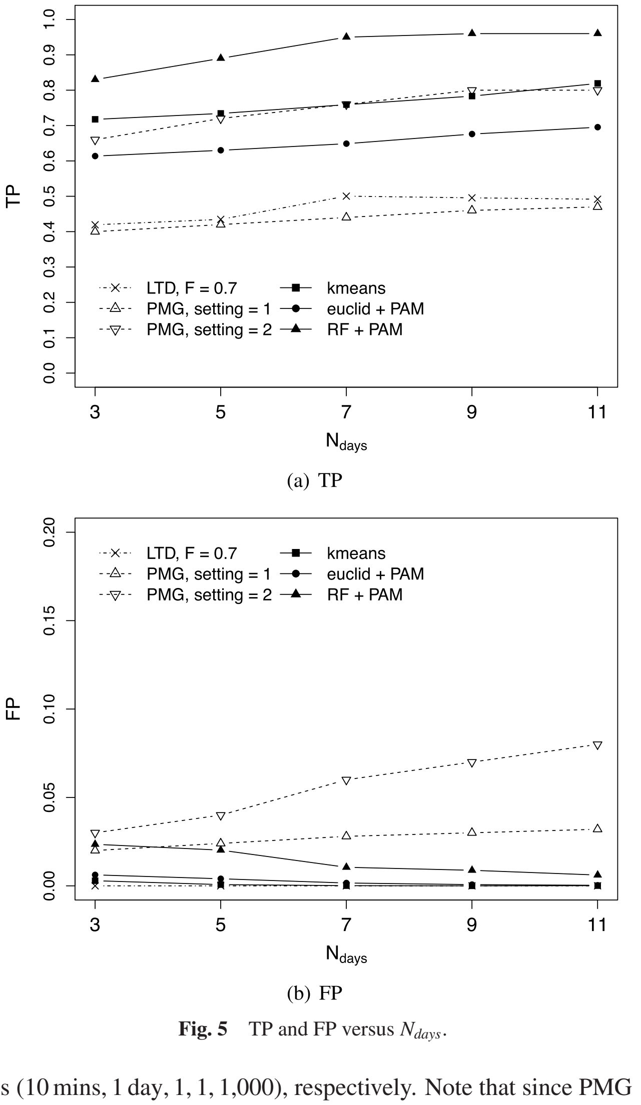


Fig. 4 Classification accuracy versus chosen feature to identify the SPITter cluster.



(a) TP

(b) FP

Fig. 5 TP and FP versus N_{days}.

is (10 mins, 1 day, 1, 1, 1,000), respectively. Note that since PMG tries to detect SPIT calls and not callers when the call is going to be established, we regard a caller as a SPITter once PMG detects a SPIT call.

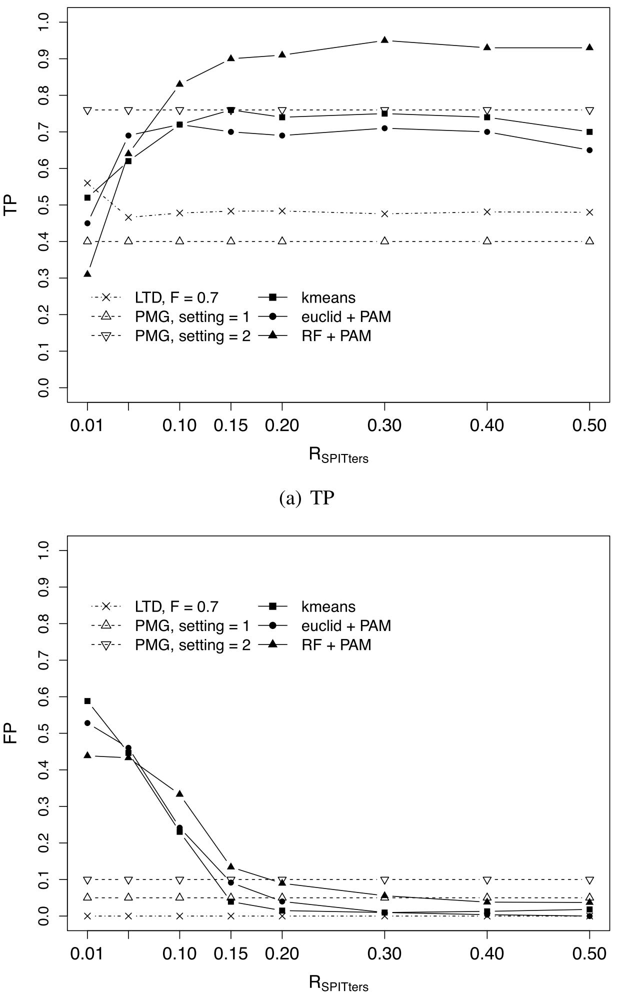
Figure 5 (a) shows the TP with varying collection periods (N_{days}). From Fig. 5 (a), we can see that TP slightly and gradually improves as N_{days} becomes longer, regardless of schemes. This result is intuitive and understandable since it has more chance to trap SPITters as the collection period gets longer. We also find that RF dissimilarity gives better TP than Euclidean distance by comparing RF + PAM, Euclid. + PAM and k-means. This is because Euclidean distance does not consider the importance of features in finding dissimilarity. Also the performance of PMG, which is a threshold-based scheme, is sensitive to threshold settings. Since PMG has to select as many as five thresholds and parameters, it is very difficult to obtain the optimal setting. LTD achieves nearly 50% TP when $N_{days} = 7$ and does not improve anymore. This is because LTD detects the callers whose ST and WT both deviate from normal and thus it cannot detect the SPITters with colluding accounts since they try to approach the values of ST which are close to those of legitimate callers.

Figure 5 (b) shows FP with varying collection period N_{days}. From Fig. 5 (b), we can see that PMGs, especially setting 2, get worse as the collection period gets longer. This is because we judge callers as SPITters once a call is judged as SPIT and thus it bring about FP as N_{days} becomes longer. It is also found that the FP of our scheme gradually decreases. In contrast to PMG, the longer the collection period gets, the more accurate classification can be executed in our scheme. This is the same reason why the TP of our scheme gets better.

From Figs. 5 (a) and (b), we can say that our scheme tries to separate legitimate callers and SPITters by dissimilarity and that it works well even though there exist low-frequency SPITters or SPITters with colluding accounts.

5.3.3 Classification Accuracy versus $R_{SPITters}$

In this section, we show the classification accuracy against $R_{SPITters}$. Since our methods rely on how legitimate callers and SPITters resembles each other, classification accuracy depends on how much SPITters accounts for inspected callers. In addition, as we stated before, the exact ratio of SPITters to inspected callers cannot be expected. Azad et al. vary $R_{SPITters}$ = 10%, 20% and 30% in Ref. [11] while Chaisamran et al. vary $R_{SPITters}$ from 1% to 10% in Ref. [29]. In order to clarify the classification accuracy against $R_{SPITters}$, we vary $R_{SPITters}$ from 1% to 50% and evaluate the classification accuracy. In this evaluation, we use not only the Reality Mining dataset but also the Nodobo dataset [28] which consists of 27 students' smartphone call logs in a Scottish state high school. This is because we cannot evaluate the classification accuracy when $R_{SPITters} < 10\%$ since the Reality Mining dataset involves only 90 callers' log. Therefore, we randomly sample $100(1 - R_{SPITters})$ legitimate callers out of 117 (= 90 + 27) persons from both datasets. **Figures 6** (a) and (b) show TP and FP versus $R_{SPITters}$, respectively. From Figs. 6 (a) and (b), we can see that the classification accuracy of our schemes gets better as $R_{SPITters}$ increases and the classification accuracy with RF + PAM outperforms when $R_{SPITters} \geq 20\%$. On the other hand, for $R_{SPITters} < 15\%$, TP in our schemes get worse as $R_{SPITters}$ becomes lower. From Fig. 6 (b), the tendency can be also seen for FP. We consider there are two reasons why our schemes get worse against low $R_{SPITters}$. The first reason is that classification itself cannot cluster well since the most of callers are legitimate when $R_{SPITters}$ is low. The second reason is that the representative feature mistakenly identifies the SPITter cluster when the chosen SPITters are only sophisticated SPITters and this brings about severe TP and FP. This situation can occur when both $N_{callers}$ and $R_{SPITters}$ are too low or few. When $R_{SPITters} = 0.01$, only one SPITter ex-



(a) TP

(b) FP

Fig. 6 TP and FP versus $R_{SPITters}$.

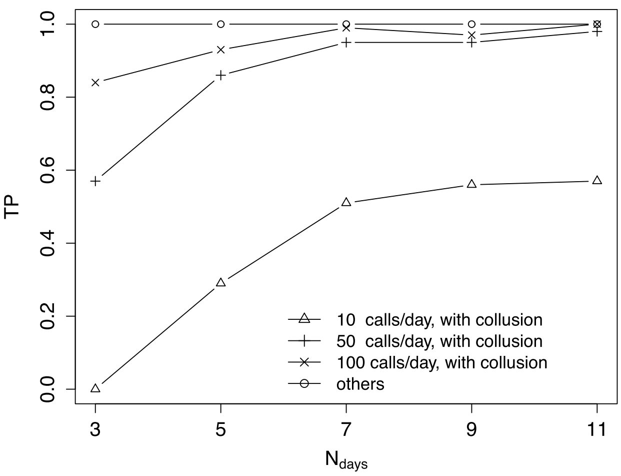


Fig. 7 TP versus SPITter types.

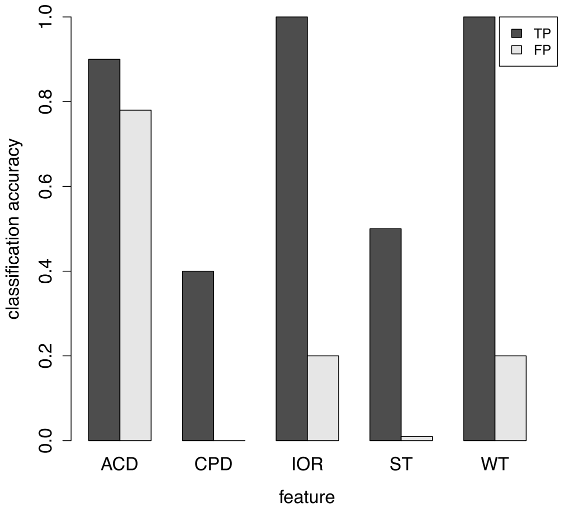


Fig. 8 Classification accuracy with single feature.

ists in the inspected callers since we use $N_{callers} = 100$. We cannot conclude whether our scheme is robust against low $R_{SPITters}$ and bigger $N_{callers}$ since we do not have a sufficient number of legitimate callers' call log.

5.3.4 Classification Accuracy versus SPITter Types

We assume ten SPITter types discussed in Section 2, which consists of low-rate SPITters (as low as 10 calls per day) to high-rate SPITters (as much as 1,000 calls per day) and with collusions or without collusions. We clarify how our scheme identifies each SPITter model. In this simulation, we use RF + PAM as the core algorithm. **Figure 7** indicates TP versus each SPITter model. In Fig. 7, we merged the result of seven types as "others." which are SPITters without collusions and high call rate SPITters ($f_{\text{CPD}} \geq 500$) with collusions, since they are correctly identified without fail. On the other hand, low-rate SPITters ($f_{\text{CPD}} \in [10, 100]$) with colluding accounts are difficult to be detected as SPITters but the TP gradually improves as time goes on.

5.3.5 Effectiveness of Each Feature

We clarify how each feature affects the ability to classify legitimate callers and SPITters. In order to clarify the effectiveness of each feature, we only use single features to represent callers. We choose k-means as the core algorithm. **Figure 8** shows the classification accuracy with each single feature. We look into the classification accuracy of each feature. Although f_{ACD} traps most SPITters, it traps legitimate callers as well. From this result, we can say that most legitimate callers do not call for a long enough duration to clearly separate SPITters. f_{CPD} and f_{ST} are less effective to detect SPITters but they bring very low FP. This is because they both represent the characteristics of ordinary people. Ordinary callers do not call so frequently and f_{ST} characterizes that legitimate ones spend most of their talking time with intimates. We show that sophisticated SPITters can imitate the ST property but it does not mean that the actual legitimate callers are not identified correctly as well. Finally we discuss the results of f_{IOR} and f_{WT}. They give us similar characteristics: both identify SPITters without fail but mistakenly identify 20% of legitimate callers as SPITters. This is because they represent the characteristic of SPITters as opposed to f_{CPD} and f_{ST}. From Fig. 8, we can see that good classification accuracy cannot be achieved with only a single feature and thus this can be the reason why multiple features have to be considered.

5.4 Calculation Complexity

We finally discuss the calculation time in our methods. In this evaluation, we vary $N_{callers}$ between 100 and 100,000 and measure

Table 7 Required time in our methods.

	Dissimilarity		Clustering	
# of callers	RF	Euclid.	PAM	*k*-means
100	1.6E−1 s	2.4E−4 s	8.7E−4 s	6.4E−4 s
1,000	3.1 s	6.7E−3 s	5.3E−2 s	1.5E−3 s
10,000	1.5E+2 s	8.3E−1 s	6.6 s	9.8E−3 s
100,000	9.4E+4 s	1.2E+1 s	4.6E+2 s	1.2E−1 s

the consumed time in calculating the dissimilarity and clustering. Our algorithms mainly consume the most of time to calculate dissimilarity and clustering. **Table 7** shows the calculation time of our scheme. From Table 7, we can see that the required time follows the time complexity for PAM ($O(N^2_{callers})$) and *k*-means ($O(N_{callers})$), respectively. In addition, RF has the highest calculation time and takes about three order of magnitude than Euclidean distance calculation. RF takes about three hours to find the dissimilarity among 100,000 callers. This might be a problem and we have to consider a calculation reduction algorithm if we consider a larger VoIP/SIP service provider.

6. Conclusion

In this paper, we have proposed an unsupervised SPIT callers detection with a clustering algorithm. Our method turns complex threshold setting and training problems into clustering the callers and identifying the cluster. In contrast to conventional schemes, we use the features to find the dissimilarity among callers and this avoids threshold tuning and the training phase. By computer simulation, we show that our scheme using RF dissimilarity and PAM clustering outperforms the conventional schemes by means of classification accuracy when SPITters account for more than 20% of inspected callers accuracy against our dataset. We also show that our scheme can tolerate as many as 100,000 callers using an off-the-shelf computer.

Acknowledgments This work is partly supported by the Grant in Aid for Scientific Research (No.26420369) from the Ministry of Education, Sport, Science and Technology, Japan.

References

[1] Shin, D., Ahn, J. and Shim, C.: Progressive Multi Gray-leveling: A Voice Spam Protection Algorithm, *IEEE Network*, Vol.20, No.5, pp.18–24 (2006).

[2] Falomi, M., Garroppo, R. and Niccolini, S.: Simulation and Optimization of SPIT Detection Frameworks, *IEEE Global Telecommunications Conference* (*GLOBECOM*), pp.2156–2161 (2007).

[3] Sengar, H., Wang, X. and Nichols, A.: Call Behavioral Analysis to Thwart SPIT Attacks on VoIP Networks, *Security and Privacy in Communication Networks*, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol.96, pp.501–510, Springer Berlin Heidelberg (2012).

[4] Keromytis, A.D.: A comprehensive survey of voice over IP security research, *IEEE Communications Surveys & Tutorials*, Vol.14, No.2, pp.514–537 (2012).

[5] Bai, Y., Su, X. and Bhargava, B.: Adaptive Voice Spam Control with User Behavior Analysis, *IEEE International Conference on High Performance Computing and Communications* (*HPCC*), pp.354–361 (2009).

[6] Wang, F., Feng, M. and Yan, K.: Voice Spam Detecting Technique Based on User Behavior Pattern Model, *8th IEEE International Conference on Wireless Communications, Networking and Mobile Computing* (*WiCOM*), pp.1–5 (2012).

[7] Breiman, L.: Random Forests, *Machine learning*, Vol.45, No.1, pp.5–32 (2001).

[8] Balasubramaniyan, V.A., Mustaque, A. and Haesun, P.: CallRank: Combating SPIT using call duration, social networks and global reputation, *Conference on Email and Anti-Spam* (*CEAS*), pp.1–8 (2007).

[9] Kusumoto, T., Chen, E.Y. and Itoh, M.: Using Call Patterns to Detect Unwanted Communication Callers, *IEEE/IPSJ International Symposium on Applications and the Internet* (*SAINT*), pp.64–70 (2009).

[10] Chaisamran, N., Okuda, T., Blanc, G. and Yamaguchi, S.: Trust-Based VoIP Spam Detection Based on Call Duration and Human Relationships, *IEEE/IPSJ International Symposium on Applications and the Internet* (*SAINT*), pp.451–456 (2011).

[11] Azad, M.A. and Morla, R.: Caller-REP: Detecting unwanted calls with caller social strength, *Computers & Security*, Vol.39, Part B, pp.219–236 (2013).

[12] Quittek, J., Niccolini, S., Tartarelli, S., Stiemerling, M., Brunner, M. and Ewald, T.: Detecting SPIT Calls by Checking Human Communication Patterns, *IEEE International Conference on Communications* (*ICC*), pp.1979–1984 (2007).

[13] Hai, H., Hong-Tao, Y. and Xiao-Lei, F.: A SPIT Detection Method Using Voice Activity Analysis, *International Conference on Multimedia Information Networking and Security* (*MINES*), Vol.2, pp.370–373 (2009).

[14] Lentzen, D., Grutzek, G., Knospe, H. and Porschmann, C.: Content-Based Detection and Prevention of Spam over IP Telephony - System Design, Prototype and First Results, *IEEE International Conference on Communications* (*ICC*), pp.1–5 (2011).

[15] Strobl, J., Mainka, B., Grutzek, G. and Knospe, H.: An efficient search method for the content-based identification of telephone-SPAM, *IEEE International Conference on Communications* (*ICC*), pp.2623–2627 (2012).

[16] Yang, W. and Judge, P.: VISOR: VoIP Security Using Reputation, *IEEE International Conference on Communications* (*ICC*), pp.1489–1493 (2008).

[17] Bokharaei, H., Sahraei, A., Ganjali, Y., Keralapura, R. and Nucci, A.: You Can SPIT, But You Can't Hide: Spammer Identification in Telephony Networks, *IEEE INFOCOM*, pp.41–45 (2011).

[18] Amanian, M., Moghaddam, M.H.Y. and Roshkhari, H.K.: New method for evaluating anti-SPIT in VoIP networks, *International eConference on Computer and Knowledge Engineering* (*ICCKE*), pp.374–379 (2013).

[19] Toyoda, K. and Sasase, I.: SPIT callers detection with unsupervised Random Forests classification, *IEEE International Conference on Communications* (*ICC*), pp.2068–2072 (2013).

[20] MacQueen, J.B.: Some methods for classification and analysis of multivariate observations, *Proc. 5th Berkeley Symposium on Mathematical Statistics and Probability*, Vol.1, pp.281–297 (1967).

[21] Kaufman, L. and Rousseeuw, P.J.: *Finding Groups in Data, An Introduction to Cluster Analysis*, John Wiley & Sons (2009).

[22] Shi, T. and Horvath, S.: Unsupervised Learning with Random Forest Predictors, *Journal of Computational and Graphical Statistics*, Vol.15, No.1, pp.118–138 (2006).

[23] Ng, R.T. and Han, J.: Efficient and Effective Clustering Methods for Spatial Data Mining, *Proc. 20th International Conference on Very Large Data Bases* (*VLDB*), pp.144–155 (1994).

[24] Liaw, A. and Wiener, M.: Classification and Regression by randomForest, *R News*, Vol.2, No.3, pp.18–22 (2002). (online), available from ⟨http://CRAN.R-project.org/doc/Rnews/⟩.

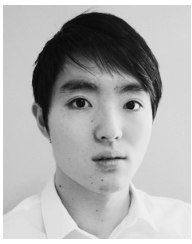
[25] Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M. and Hornik, K.: cluster: Cluster Analysis Basics and Extensions (2013).

[26] Eagle, N. and Pentland, A.: Reality Mining: Sensing Complex Social Systems, *Personal and Ubiquitous Computing*, Vol.10, No.4, pp.255–268 (2006).

[27] Blondel, V.D., Esch, M., Chan, C., Clerot, F., Deville, P., Huens, E., Morlot, F., Smoreda, Z. and Ziemlicki, C.: Data for Development: The D4D challenge on mobile phone data, *arXiv preprint arXiv:1210.0137* (2012).

[28] Bell, S., McDiarmid, A. and Irvine, J.: Nodobo: Mobile phone as a software sensor for social network research, *IEEE Vehicular Technology Conference* (*VTC Spring*), pp.1–5 (2011).

[29] Chaisamran, N., Okuda, T. and Yamaguchi, S.: Trust-based VoIP Spam Detection based on Calling Behaviors and Human Relationships, *Journal of Information Processing*, Vol.21, No.2, pp.188–197 (2013).



Kentaroh Toyoda was born in 1988. He received his M.S. degree from Keio University in 2013. He is a Ph.D. student and a research assistant at Keio University. His research interest is security & privacy. He is a member of IEEE, IPSJ, and IEICE.

Iwao Sasase was born in Osaka, Japan in 1956. He received the B.E., M.E., and D.Eng. degrees in Electrical Engineering from Keio University, Yokohama, Japan, in 1979, 1981 and 1984, respectively. From 1984 to 1986, he was a Post Doctoral Fellow and Lecturer of Electrical Engineering at the University of Ottawa, ON, Canada. He is currently a Professor of Information and Computer Science at Keio University, Yokohama, Japan. His research interests include modulation and coding, broadband mobile and wireless communications, optical communications, communication networks and information theory. He has authored more than 270 journal papers and 400 international conference papers. He granted 41 Ph.D. degrees to his students in the above field. Dr. Sasase received the 1984 IEEE Communications Society (ComSoc) Student Paper Award (Region 10), 1986 Inoue Memorial Young Engineer Award, 1988 Hiroshi Ando Memorial Young Engineer Award, 1988 Shinohara Memorial Young Engineer Award, 1996 Institute of Electronics, Information, and Communication Engineers (IEICE) of Japan Switching System Technical Group Best Paper Award, and WPMC 2008 Best Paper Award. He served as President-in-Elect of the IEICE Communications Society (2012–2013). He was Board of Governors Member-at-Large (2010-2012), Japan Chapter Chair (2011–2012), Director of the Asia Pacific Region (2004–2005), Chair of the Satellite and Space Communications Technical Committee (2000–2002) of IEEE ComSoc., Vice President of the Communications Society (2004–2006), Chair of the Network System Technical Committee (2004–2006), Chair of the Communication System Technical Committee (2002–2004) of the IEICE Communications Society, Director of the Society of Information Theory and Its Applications in Japan (2001–2002). He is Fellow of IEICE, and Senior Member of IEEE, Member of the IPSJ.