Evaluating Classifiers for Mobile-Masquerader Detection*

Oleksiy Mazhelis[1], Seppo Puuronen[1], and Mika Raento[2]

[1] University of Jyväskylä, Finland
{mazhelis,sepi}@it.jyu.fi
[2] University of Helsinki and Helsinki Institute for Information Technology, Finland
Mika.Raento@cs.Helsinki.FI

Abstract. As a result of the impersonation of a user of a mobile terminal, sensitive information kept locally or accessible over the network can be abused. The means of masquerader detection are therefore needed to detect the cases of impersonation. In this paper, the problem of mobile-masquerader detection is considered as a problem of classifying the user behaviour as originating from the legitimate user or someone else. Different behavioural characteristics are analysed by designated one-class classifiers whose classifications are combined. The paper focuses on selecting the classifiers for mobile-masquerader detection. The selection process is conducted in two phases. First, the classification accuracies of classifiers are empirically evaluated, and inaccurate classifiers are excluded. After that, the accuracies of different classifier combinations are explored, and the combination with the best classification accuracy is identified. The experimental results suggest that, in order to achieve better accuracy, the individual classifiers with both high classification accuracy and a small number of non-classifications need to be selected.

1 Introduction

An impersonation of the user of a contemporary mobile terminal, such as a smart-phone or PDA, may result in an abuse of critical private or corporate information. Impersonating the legitimate user (later called *user*) to obtain an unauthorised access to sensitive data or services authorised for that user is referred to as the *masquerade attack.* In order to resist such attack, the so-called *detective* security means [1] can be implemented (in addition to preventive security means) to perform *masquerader detection.* During last years, great efforts have been devoted to the problem of detecting masquerade attacks, e.g. [2–5], also in the context of mobile terminals [6].

In this paper, the problem of mobile-masquerader detection is approached as an anomaly detection problem [2]. To detect anomalies, a user model is learnt through the monitoring of the user behaviour during the so-called learning phase, and the learnt

* This work was partly supported by the COMAS Graduate School of the University of Jyväskylä. The Context project was funded by the Academy of Finland under the PROACT research program. The authors would like to thank Hannu Toivonen, as well as anonymous reviewers for valuable comments and suggestions.

model is stored in a *user profile*. The detection may be performed by matching a currently observed behaviour of a claimant (a person whose identity claim is being verified) against the model. Multiple behavioural characteristics have been proposed as potentially useful in masquerader detection [6, 7, 3]. Recently, a set of characteristics for mobile-masquerader detection have been suggested [8]; however, an empirical evidence indicating how well these characteristics can be used in practise has not been provided so far.

We formulate the problem of anomaly detection as a one-class classification problem [9], where a claimant's behaviour, represented by a set of feature values, is classified as belonging to the *user class* reflecting the behaviour of the user, or not. In the latter case, the decision is made that the behaviour belongs to the *impostor class* summing up the behaviours of all other individuals but the user.

In anomaly detection, one-class classifiers are often based on probability estimation [10, 2, 4], and they often analyze the whole set of features simultaneously. However, due to difficulties with learning when the features are lumped into a single high-dimensional vector, difficulties with the normalization of features having different physical meaning, and partial availability of the feature values at the time the detection is performed, the use of an alternative approach based on combining classifiers is justified [11]. Following this approach, the features can be divided into subgroups processed by individual one-class classifiers. Each of them is aimed at classifying the current values of assigned features. By employing a *combining rule*, the final classification is produced based on the classifications of the individual classifiers.

In this paper, we empirically evaluate one-class classifiers to be used in mobile-masquerader detection. For the purposes of the evaluation, the dataset describing the behaviour of nine mobile users was employed. The selection of features and classifiers was constrained by the attributes available in this dataset. The authors are not aware of any other empirical study on mobile-masquerader detection, where more than one behavioural aspect is taken into account.

The evaluation was conducted in two phases. In the first phase, classification accuracies of classifiers were evaluated, and the classifiers with low accuracy were filtered. In the second phase, we investigated which of the remaining classifiers are reasonable to use in combination so that the accuracy of final classification would increase. For this, the accuracies of different classifier combinations were compared. When selecting the classifiers to add to the combination, a variety of criteria can be used; some of them were tested. The best results were achieved, when the classifiers selected had both high classification accuracy and a low number of non-classifications.

The paper is organised as follows. In the next Section, the features employed are described, the design of individual classifiers is discussed, and the details of the utilised combining scheme are provided. The employed dataset is described in Section 3. The experimental settings and the results of experiments are presented in Section 4. Finally, in Section 5, the experimental findings and their limitations are discussed, and the topics for further work are outlined.

2 Design of individual classifiers and their combination

In this section, the measures employed for masquerader detection and the design of individual classifiers to process these measures are considered, and the scheme for combining individual classifications is described.

2.1 Design of individual classifiers

In our approach, the anomaly detection problem is seen as a classification problem, where the object Z (claimant) is to be classified as belonging either to the user class ($Z \in C_U$) or to the class of impostors ($Z \in C_I$), but not to both of them, i.e. $\{Z|Z \in C_U\} \cup \{Z|Z \in C_I\} = \emptyset$. The object Z is represented by the set of n_f features $\{x_1, \ldots, x_{n_f}\}$ from the feature space $\mathcal{X}$, and is analysed by the set of R individual classifiers. Classifier i takes as input the observation vector $\mathbf{x}_i \equiv (x_1^{(i)}, \ldots, x_{n_{f_i}}^{(i)})$, $x_j^{(i)} \in \mathcal{X}_i \subset \mathcal{X}$.

The classification process consists of the learning and the classification phases. In the learning phase, using a training set $\mathcal{DS}_T$, the classifier i learns the set of parameters Θ_i constituting the *model* of the classifier. $\mathcal{DS}_T$ includes the vectors of feature values of the user: $\mathcal{DS}_T = \{((x_1, \ldots, x_{n_f})_j, y_j)|j = 1, \ldots, |\mathcal{DS}_T|\}$, where $y_j = C_U$ is the class label. In the classification phase, the learnt model is used to classify into the user class or the impostor class unlabeled observation vectors from a dataset $\mathcal{DS}_C = \{((x_1, \ldots, x_{n_f})_j)|j = 1, \ldots, |\mathcal{DS}_C|\}$.

Observation vectors of individual classifiers are initialised with available feature values gathered by employing a sliding window $[\tau_1, \tau_2]$ of the length $l_\tau = \tau_2 - \tau_1$ (determining the time interval, within which the feature values are collected), with an increment for the window δ_τ. Given an unlabeled observation vector $\mathbf{x}_i$, each classifier outputs the individual classification $u_i = u_i(\mathbf{x}_i, \Theta_i)$ indicating how likely $Z \in C_U$. Below, the descriptions of the employed features and the individual classifiers are provided.

In [8], the authors considered personality factors in order to identify suitable characteristics and measures for mobile-masquerader detection. They suggest that, while personality factors are latent, they are reflected in different *aspects* of user *behaviour* and *environment*. The characteristics describing these aspects can be directly observed. It is hypothesized that the superposition of characteristics describing one's behaviour and environment is also individual, and can be used to verify the identity of a person.

To measure quantitatively the characteristics of user behaviour and environment, one or more appropriate observable variables, or *measures* should be assigned to each of them. A list of tentative characteristics and measures has been proposed; the interested reader may consult [8] for the detailed description of these characteristics. Some of the measures described in [8] are not available in the dataset (described in Section 3) which was employed in the experiments. The available measures were assigned as features to individual classifiers. To each of the measures, an individual classifier was assigned as described below.

Type of program or service evoked. Active applications evoked by the user are registered in the dataset. This measure is referred to as *active applications* (ACT_APP).

For each application j, the classifier based on active applications maintains a counter a_{app_j} that stores the number of times the user evokes the application. The probability of an application j being evoked out of m applications is approximated as $\hat{P}(\mathrm{app}_j|U) = (a_{\mathrm{app}_j} + 1)/(\sum_m a_{\mathrm{app}_m} + 1)$. Assuming the independence of consequent application evocations, the probability of application evocations within a time window $[\tau_1, \tau_2]$ is approximated as

$$\hat{P}(\mathrm{app}_{i-n_{app}+1}, \ldots, \mathrm{app}_i|U) = \prod_{j=i-n_{app}+1}^{i} \hat{P}(\mathrm{app}_j|U), \tag{1}$$

where app_i is the last application evoked within the time window, and n_{app} is the average number of applications evoked within the window. The application evocations within previous window(s) can be taken into account if needed (the same is valid for the other classifiers, too). Given the current active applications to be classified, the classifier outputs the classification $u_i = \hat{P}(\mathrm{app}_{i-n_{app}+1}, \ldots, \mathrm{app}_i|U)$.

Sequence of cells traversed. The dataset records the identifiers of the cells (Cell IDs) wherein the mobile terminal is registered. The information about consecutive Cell IDs can be utilised in order to create the *sequences of cells traversed* measure ($MOVE$).

The model of the assigned classifier includes the matrix of cell-to-cell transition probabilities, and, in fact, represents a first-order Markov model. An element $a_{\mathrm{cell}_i\ \mathrm{cell}_j}$ of the matrix is a counter that stores the number of times the terminal's Cell ID changed from cell i to cell j. The matrix values are used in approximating the probability $\hat{P}(\mathrm{cell}_j|\mathrm{cell}_i, U)$ of a handover:

$$\hat{P}(\mathrm{cell}_j|\mathrm{cell}_i, U) = \frac{a_{\mathrm{cell}_i\ \mathrm{cell}_j} + 1}{\sum_m a_{\mathrm{cell}_i\ \mathrm{cell}_m} + n_{neighbour\ i}}, \tag{2}$$

where cell_m are the cells to which traversals from cell_i were registered, and $n_{neighbour\ i}$ is the number of such cells. Given the parameters l_τ and δ_τ of sliding windows, the average number of handovers n_{ho} within a window was estimated. Assuming the independence of consequent handovers, the probability of a sequence of cell changes within a time window $[\tau_1, \tau_2]$ was approximated as

$$\hat{P}(\mathrm{cell}_{i-n_{ho}}, \ldots, \mathrm{cell}_i|U) = \prod_{j=i-n_{ho}}^{i-1} \hat{P}(\mathrm{cell}_{j+1}|\mathrm{cell}_j, U), \tag{3}$$

where cell_i is the last cell registered within the time window. In the classification phase, given the current route to be classified, the classifier outputs the classification $u_i = \hat{P}(\mathrm{cell}_{i-n_{ho}}, \ldots, \mathrm{cell}_i|U)$.

Speed of move. The speed of terminal movements is not available in the dataset. However, the timestamps of the Cell ID records can be used to estimate the time the terminal spends in a cell, which, in turn, can be used to roughly estimate the terminal's speed (e.g. in terms of "cell per second") referred as *speed* ($SPEED$) measure.

The speed-based classifier was constructed as follows. For each cell, the user speed is modelled separately based on the empirical distribution of the user speed in this cell. The value of the speed can be approximated as a ratio of the length of the user's path within the cell to the time the user spent in the cell. The time spent in the cell is estimated as the time interval $[\tau_{ho1}, \tau_{ho2}]$ between consequent handovers. It is assumed that the user follows the same path within the cell; hence, the length of the path is assumed constant and is omitted from the expression for speed calculation, i.e. the speed in the cell is estimated as $v_{\text{cell}_i} = 1/(\tau_{ho2} - \tau_{ho1})$ (in "cell per second"). Only smaller values ($\tau^{stay} < 11$ minutes) are processed by the speed-based classifier, while greater values are assumed to indicate that the terminal is not moving. Using the accumulated empirical distribution function (EDF) of v_{cell_i}, the probability density $p(v_{\text{cell}_i})$ of the current speed for cell i is evaluated by using k-Nearest-Neighbours method [12]. Assuming independence of the speed in subsequent cells, the likelihood of a user speed within a time window $[\tau_1, \tau_2]$ is approximated as

$$L_{speed}(\text{cell}_{i-n_c+1}, \ldots, \text{cell}_i|U) = \prod_{j=i-n_c+1}^{i} p(v_{\text{cell}_j}), \tag{4}$$

where cell_i is the last cell registered within the time window, and n_c is the average number of cell changes within a window. Given the current speed values, the classifier outputs the classification $u_i = L_{speed}(\text{cell}_{i-n_c+1}, \ldots, \text{cell}_i|U)$.

Locations where prolonged stops were made. The information about the Cell IDs and the time spent in cells can be used to identify those locations (in terms of Cell IDs) where the terminal stays for a relatively long period of time. This measure is named *places visited* ($PLACES$).

The design of the classifier based on this measure is similar to the design of the classifier based on active applications. The difference is that the locations (Cell IDs) of prolonged stops, as defined in the description of the speed-based classifier, are taken as input instead of the application identifiers.

Temporal interval between two consecutive evocations of a program or service of a same type. In the dataset, the evocations of services (calls and SMS) are recorded and time-stamped; using this information, the intervals between evocations of these services can be evaluated. These measures will be referred to as *arrival of calls* (ARR_CALL) and *arrival of SMS* (ARR_SMS).

For both services, classifiers of the same type are employed. These classifiers analyse the mean inter-arrival time $\overline{\tau}^{ia}$ within a window $[\tau_1, \tau_2]$ for a given service type. The value of $\overline{\tau}^{ia}$ is calculated as $\overline{\tau}^{ia} = \frac{1}{n_{sa}} \sum_{j=i-n_{sa}+1}^{i} \tau_j^{ia}$, where τ_i^{ia} is the last inter-arrival time observed within the window, and n_{sa} is the average number of service arrivals within windows. Using the accumulated EDF of the $\overline{\tau}^{ia}$ values, the probability density $p(\overline{\tau}^{ia})$ of the current mean inter-arrival time is evaluated using k-Nearest-Neighbours method. Given the current inter-arrival time values, the classifier outputs the classification $u_i = p(\overline{\tau}^{ia})$.

Temporal lengths of actions. In the dataset, the durations of calls are recorded; they are used as a *call duration* (DUR_CALL) measure. The design of the classifier based on call duration time is similar to the design of the previous classifier. The difference is that the durations of calls are taken as input instead of the service inter-arrival times.

Address information of the people contacted. Phone numbers contacted via calls or SMS are available in the dataset. This measure is referred to as *contact numbers*

($CONT\_NUM$). Besides, the identifiers (MAC-addresses) of neighbouring Bluetooth-devices are logged in the dataset and are employed as *neighbouring Bluetooth devices* (BT_DEV) measure.

The classifier based on the phone numbers of contacted people, and the classifier based on the MAC-addresses of neighbouring Bluetooth devices are designed similarly to the classifier based on active applications. The difference is that contact numbers or MAC-addresses, respectively, are taken as input instead of the application identifiers.

2.2 Combining individual classifications

Combining the classifications produced by several classifiers has been extensively explored as a mean to compensate the weaknesses of individual classifiers. Different combining rules have been investigated [11, 13], and it has been shown that combining may result in a significant reduction of classification errors [13, 14]. However, in combining one-class classifiers, where only the knowledge regarding one class is available, relatively few rules can be used. Among them are different modifications of voting rules as investigated by Xu et al. [11]. Tax [9] reported the applicability of mean vote, mean weighted vote, product of weighted votes, mean of the estimated probabilities, and product combination of probabilities as combining rules for one-class classifiers. In [15], the mean of the estimated probabilities (MP) rule was justified to be among the most suitable ones in the context of mobile masquerader detection, and an improved version of it (modified MP rule) was proposed. This modified MP rule is adopted in the experiments as a scheme for combining individual classifications. Below, the details of this rule are provided.

The modified MP rule assumes that each classifier i outputs its classification as an estimation of the probability density function (pdf) for the user class $p(\mathbf{x}_i|C_U)$. Given R classifiers to be combined, the rule represents the average of the classifier confidences u_i^c:

$$u_{mc}(\mathbf{x}_1, \ldots, \mathbf{x}_R) = R^{-1} \sum_{i=1}^{R} u_i^c(p(\mathbf{x}_i|C_U)), \tag{5}$$

where u_i^c reflects the degree of the classifier's confidence in the hypothesis that an object Z belongs to the user class. The confidence values can be calculated as [15]:

$$u_i^c(p(\mathbf{x}_i|C_U)) = \frac{1}{1 + \exp\left(-\ln \frac{p(\mathbf{x}_i|C_U)}{\overline{p}(\mathbf{x}_i|C_U)}\right)} = \frac{p(\mathbf{x}_i|C_U)}{p(\mathbf{x}_i|C_U) + \overline{p}(\mathbf{x}_i|C_U)}, \tag{6}$$

where $\overline{p}(\mathbf{x}_i|C_U)$ is the mean value of the estimated probability $p(\mathbf{x}_i|C_U)$. This mean value is equal to the probability of a random variable uniformly distributed in the feature space $\mathcal{X}_i$.

The final classification result using the modified MP combining rule is made by comparing the obtained u_{mc} value with a threshold t_{mc}:

$$\begin{aligned} &\text{Decide} \quad Z \in C_U \quad \text{if} \quad u_{mc} \geq t_{mc}, \\ &\text{otherwise decide} \quad Z \in C_I. \end{aligned} \tag{7}$$

3 Dataset

The dataset used in this study was obtained from the Context project at the University of Helsinki and was gathered using the ContextPhone software [16]. ContextPhone runs in the background on Series 60 smartphones, collects data and sends it to a server automatically, without the need for user interaction. The data describes the users' movements (GSM cell changes), phone usage (phone profile, application use, idle and active time, charger), physical social interaction (bluetooth environment) and mobile phone communication (phone calls and text messages).

The data comes from two field studies conducted to test ContextContacts, a social awareness service [17]. The first study was done with a family of four: mother and three children aged 10 to 16. The second group consisted of five high-school students, aged 16 to 18, who ran a small company together. Both studies lasted approximately three months. All subjects were Finns living in the greater Helsinki area. The anonymized version of the dataset is available from `http://www.cs.helsinki.fi/group/context/data/`.

4 Experimental results

In this section, the experimental settings are described, and the results of the experiments are presented.

4.1 Experimental settings

The experiments were conducted in two phases. The purpose of the first phase was to evaluate the classification accuracy of the individual classifiers and to exclude the classifiers with low accuracy. In turn, the experiments in the second phase were aimed at identifying the classifiers, the combination of which would allow the final classification accuracy to be improved, as compared with the accuracies of individual classifiers. To identify the individual classifiers to be combined, the classifiers were ranked according to several criteria. After that, the best classifiers (according to the ranks) were added one by one to the classifier combination, and the resulting classification accuracy was estimated.

The holdout cross-validation [18] was used in the experiments to assess the accuracies. The model of each classifier was learnt using the training data-set $\mathcal{DS}_T$, and was subsequently used to classify the instances of the classification data-set $\mathcal{DS}_C$. In general, the classification data-set $\mathcal{DS}_C$ should include both the instances originated from the user and the instances originated from impostors. However, since the data originated from impostors was not available, the other users' data was employed as the impostor's data.

For each user, the data was split into two parts in the relation $2:1$ commonly used in classifier evaluation [18]. The first part formed the training data-set $\mathcal{DS}_T$ that was used to learn the model for this user only. The second part was included into the classification data-set $\mathcal{DS}_C$ that was used by all the classifiers within the same user group in the classification phase. The sliding window of the length of $l_\tau = 1800$ seconds with the increment of $\delta_\tau = 900$ seconds was used.

In order to evaluate the accuracy of a classifier distinguishing between a user and impostors, the values of the probability of correct detection P_D and false rejection (FR) error rates P_{FR} are usually employed. A correct detection happens if an impostor is correctly classified as belonging to the impostors, and a false rejection occurs when a legitimate user is classified as an impostor. The ideal accuracy, corresponding to the values $P_D = 1$ and $P_{FR} = 0$ is extremely difficult, if at all possible, to achieve. Therefore, in practice, a trade-off between P_{FR} and P_D is set as a goal. The dependence between P_D and P_{FR} values can be represented by a so-called receiver-operating curve (ROC-curve) depicting the P_D values as a function of P_{FR}. The area under the curve (AUC) [19] was employed in the experiments, as it reflects the classifier accuracy; in general, the greater area corresponds to the classifier with the better accuracy.

Base ROC-curves, along with corresponding AUCs, reflect how accurate single classifications are, provided that the classifications can be made by the classifier. However, they do not take into account those observation vectors, for which no classification can be provided by the classifier, due to the absence of corresponding feature values in a particular window. The number of such non-classifications is different for different individual classifiers; therefore, their classification accuracies cannot be compared using such base ROC-curve and AUC.

Therefore, along with the base ROC-curve, a normalised ROC-curve and normalised AUC were introduced. A normalised ROC-curve depicts P_D^{norm} as a function of P_{FR}^{norm}, where P_D^{norm} and P_{FR}^{norm} represent respectively the normalisations of P_D and P_{FR}, wherein the cases of non-classifications are taken into account. For classifier i, the values of P_{FR}^{norm} and P_{FR}^{norm} are calculated as:

$$P_{D_i}^{norm} = P_{D_i}\, n_{C_i}/n_{C\ max}, \quad P_{FR_i}^{norm} = P_{FR_i}\, n_{C_i}/n_{C\ max}, \tag{8}$$

where n_{C_i} denotes the number of classifications made by classifier i, and $n_{C\ max}$ is the maximum number of classifications that can be made by individual classifiers or combinations thereof. Furthermore, for the values of $P_{D_i}^{norm}$ and $P_{FR_i}^{norm}$ greater than $n_{C_i}/n_{C\ max}$, we assume $P_{D_i}^{norm} = P_{FR_i}^{norm}$; this reflects the behaviour of a classifier randomly guessing whenever it is unable to make a classification. Given the value of base AUC_i, the normalised AUC can be calculated as:

$$AUC_i^{norm} = 0.5 + \frac{(n_{C_i})^2}{(n_{C\ max})^2}(AUC_i - 0.5) \tag{9}$$

Thus, the base AUC reflects how accurate a single classification is provided that the classification can be made, while the normalised AUC reflects how many accurate classifications can be made in general, taking into account both classifications made and non-classifications.

4.2 Evaluating classifiers' accuracy

In this subsection, the accuracy of individual classifiers is considered. For each user, the accuracy is evaluated individually by using base and normalised AUCs. In Table 1, the base AUCs for individual classifiers averaged over all users are presented. As can be seen from the table, the ARR_CALL and ARR_SMS classifiers had low accuracy ($\overline{AUC} \approx 0.5$), and hence they were excluded from further consideration. The low accuracy of these classifiers suggests that the service inter-arrival times are similar across different users, and hence neither of them is a good differentiator between users. On the other hand, the relatively low accuracy of the $SPEED$ classifier might be attributed to the fact that the Cell IDs, on which this classifier relies, may change sporadically [20], e.g. due to the effect of interference. As a result, the produced estimation of the speed is both biased and noisy.

Table 1. Averaged base and normalised AUCs for individual classifiers.

Classifier	ARR_CALL	ARR_SMS	DUR_CALL	PLACES	SPEED	MOVE	CONT_NUM	BT_DEV	ACT_APP
$\overline{AUC}$	0.5006	0.5007	0.5036	0.7429	0.5370	0.5775	0.8062	0.7090	0.5436
$\overline{AUC}^{norm}$	–	–	0.5003	0.5429	0.5059	0.5489	0.5176	0.5358	0.5125

The normalised AUCs for remaining seven classifiers are also provided in Table 1. As could be seen, should the classifiers be ranked according to their accuracy, the produced accuracy ranks would differ significantly depending on whether the base or normalised AUC is taken into account. Specifically, while $CONT\_NUM$ is the most accurate classifier according to the base AUC, it is rather poor according to the normalised AUC. On the contrary, the $MOVE$ classifier, while not performing well according to the base AUC, is the most accurate classifier according to the normalised AUC. This is due to the difference in the number of classifications that different classifiers are able to produce. For example, the abovementioned $MOVE$ classifier in average produced 4840 classifications per user, while the $CONT\_NUM$ classifier was able to produce in average only 1237 classifications. Consequently, for a same number of windows, more true detections can be produced by (generally less accurate) $MOVE$ classifier.

4.3 Selecting classifiers to combine

In this phase of the experiments, different combinations of classifiers were explored. The classifiers were added to the combination on a one-by-one basis. In order to select the next classifier to add, the individual classifiers were ranked according to several criteria, including:

- Classification accuracy according to the base AUC;
- Classification accuracy according to the normalised AUC;
- Heuristic accuracy measure $\overline{AUC} \times n_{C_i}$ taking into account both the classification accuracy (base AUC) and the number of classifications the classifier is able to produce.

The resulting ranks are shown in Table 2. As reflected in the table, in many cases, classifiers' ranks differ depending on the criteria used. Therefore, several configurations

Table 2. Ranks of individual classifiers.

Classifier	DUR_CALL	PLACES	SPEED	MOVE	CONT_NUM	BT_DEV	ACT_APP
Rank acc. $\overline{AUC}$	7	2	6	4	1	3	5
Rank acc. $\overline{AUC_{norm}}$	7	2	6	1	4	3	5
Rank acc. $\overline{AUC} \times n_{C_i}$	7	1	6	4	3	2	5

Table 3. Averaged base and normalised AUCs for classifier combinations.

Classifiers combined	$\overline{AUC}$	$\overline{AUC_{norm}}$
PLACES+CONT_NUM	0.7490	0.5742
PLACES+BT_DEV	0.7521	0.6080
CONT_NUM+BT_DEV	0.7316	0.5719
PLACES+MOVE	0.6119	0.5738
PLACES+MOVE+BT_DEV	0.6788	0.6397
PLACES+CONT_NUM+BT_DEV	0.7637	0.6452
PLACES+CONT_NUM+BT_DEV+ACT_APP	0.7139	0.6673
PLACES+MOVE+CONT_NUM+BT_DEV	0.6904	0.6687
PLACES+MOVE+CONT_NUM+BT_DEV+ACT_APP	0.6854	0.6849
DUR_CALL+PLACES+SPEED+MOVE+CONT_NUM+BT_DEV+ACT_APP	0.6658	0.6658

of classifier combinations were tried, and better configurations were subsequently selected. The resulting classification accuracy for different configurations is reported in Table 3.

As indicated by the normalised AUCs, the best combinations of two, three, and four classifiers are produced by adding sequentially $PLACES$, BT_DEV, $CONT\_NUM$, and $MOVE$. Thus, for two-, three-, and four-classifier combinations, the best configurations of classifiers is produced, when the classifiers are added to the combination in accordance with the heuristic accuracy measure $\overline{AUC} \times n_{C_i}$ introduced above. This suggests that the best resulting classification accuracy may be achieved, when the classifiers to be added both i) have high individual classification accuracy (manifested in the base AUC) and ii) produce few non-classifications (manifested in n_{C_i} value).

Note that the best base AUCs increase when the second and the third classifier are added, while adding subsequent classifiers reduces the resulting base AUCs. Meanwhile, the best normalised AUCs increase also when the forth and the fifth classifiers are added, and deteriorate when the sixth and the seventh are included. This indicates that the benefits from being able to make more classifications with the forth and the fifth classifier overweigh the decrease in the accuracy of separate classifications. It should be also noted that, while the normalised AUCs increase when the forth and the fifth classifiers are added, the analysis of the normalised ROC-curves reveals, that the ROC-curves for the four- and five-classifier combinations surpass the ROC-curves for the best three-classifier combination only for high values of FR rate (greater than 0.5). The three-classifier combination was also found superior by analysing partial AUCs, produced by restricting the ROC-curves to the FR rate lower than 0.4. Thus, for the applications where low values of FR rate are important, the use of the three-classifier combination may in fact be a better choice.

5 Discussion and concluding remarks

The means of masquerader detection can be used to resist unauthorised use of sensitive information accessible on mobile terminals. In this paper, the problem of mobile-masquerader detection is approached as an anomaly detection problem. The profile of normal user behaviour is built by monitoring the user over a long period of time. After that, the current user behaviour is matched against the profile, and, should significant discrepancies be found, they are flagged as potential masquerade attacks.

A number of measures have been proposed as potentially useful for mobile-masquerader detection. In this paper, the appropriateness of some of them was empirically tested by using the data describing the behaviour of nine mobile users. Individual one-class classifiers were constructed for each of the measures being verified, and the accuracy of these classifiers and combinations thereof was empirically compared. As a result of experiments, two classifiers (*ARR_CALL*, *ARR_SMS*) were found inaccurate; this suggests that the corresponding measures are not useful in mobile-masquerader detection. The accuracies of classifiers and their combinations were compared by using normalised AUCs. The best accuracy was achieved when several classifiers were combined. Furthermore, the best classifier combination was produced when the classifiers both i) produced accurate individual classifications (as reflected in the values of the base AUC) and ii) were able to make many classifications (as reflected in the average numbers of classifications made).

For smaller values of FR rate, the best accuracy was found to be produced by the combination of *PLACES*, *CONT_NUM*, and *BT_DEV* classifiers. Noteworthy, the corresponding measures describe the environment of a mobile user from different viewpoints: locations of visited places describe a user's global physical environment, the contacted numbers describe a user's global virtual (social) environment, and neighbouring Bluetooth devices describe a user's local virtual (social) environment. The involvement of several measures into the process of masquerader detection makes the detection more difficult to subvert, since several aspects of the legitimate user's environment need to be mimicked simultaneously by a masquerader. The personal data employed for masquerader detection can be collected, stored, and processed directly on the terminal, without the need to be transmitted and stored on a remote server, and, therefore, this data does not need to be disclosed to a trusted party.

Due to the use of real data in the experiments, the results are likely to be generalisable. On the other hand, due to the peculiarities of the employed dataset, used classifiers, etc., some bias may be contained in the results obtained. Below, these peculiarities are discussed.

First, in the experiments, an attempt was made to distinguish users within groups, wherein the users were acquainted with each other and, hence, shared some of the characteristics (e.g. places visited). Therefore, for some of the classifiers the task of distinguishing between users was more challenging, and their classification accuracy might be underestimated. Second, the design of some of the individual classifiers is likely to be suboptimal. For instance, the low accuracy of the *SPEED* classifier may be attributed to the inaccurate estimation of the speed that was based on the Cell ID changes; the use of a location recognition algorithm [20] to pre-process the data might improve the classifier's accuracy significantly. Third, in the experiments, it was assumed that the behaviour of impostors may be approximated by the behaviour of other users. Whether such approximation is accurate enough remains to be a question for further study.

In addition to the classification accuracy, the memory requirements and the computational overhead imposed by the proposed masquerader detection approach, need to be evaluated. Their evaluation, however, depends on the number and type of classifiers used, among other parameters. Since the current work was focused on the evaluation of individual classifiers and combinations thereof, the evaluation of the required computational resources was left for further work.

Finally, the behaviour of a limited number of users was described by the dataset, and these users may not reflect accurately the behaviour of mobile users in general. Therefore, the reported results may need to be refined in further research, using bigger datasets describing the behaviour of a larger population of mobile users.

References

1. Straub, D.W., Welke, R.J.: Coping with systems risk: Security planning models for management decision making. MIS Quarterly **22**(4) (1998) 441–469
2. Schonlau, M., DuMouchel, W., Ju, W., Karr, A., Theus, M., Vardi, Y.: Computer intrusion: Detecting masquerades. Statistical Science **16**(1) (2001) 58–74
3. Lane, T., Brodley, C.E.: An empirical study of two approaches to sequence learning for anomaly detection. Machine Learning **51**(1) (2003) 73–107
4. Shavlik, J., Shavlik, M.: Selection, combination, and evaluation of effective software sensors for detecting abnormal computer usage. In: Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining, ACM Press (2004) 276–285
5. IndrajitRay, NayotPoolsapassit: Using attack trees to identify malicious attacks from authorized insiders. In de Capitani di Vimercati, S., Syverson, P., Gollmann, D., eds.: Proceedings of ESORICS 2005. Volume 3679 of Lecture Notes in Computer Science., Springer-Verlag GmbH (2005) 231–246
6. Clarke, N.L., Furnell, S.M., Lines, B., Reynolds, P.L.: Keystroke dynamics on a mobile handset: A feasibility study. Information Management and Computer Security **11**(4) (2003) 161–166
7. Hollmen, J.: User Profiling and Classification for Fraud Detection in Mobile Communications Networks. PhD thesis, Helsinki University of Technology (2000)
8. Mazhelis, O., Puuronen, S.: Characteristics and measures for mobile-masquerader detection. In Dowland, P., Furnell, S., Thuraisingham, B., Wang, X.S., eds.: Proc. IFIP TC-11 WG 11.1 & WG 11.5 Joint Working Conference on Security Management, Integrity, and Internal Control in Information Systems, Springer Science+Business Media (2005) 303–318
9. Tax, D.: One-class classification. Ph.D. thesis, Delft University of Technology (2001)
10. Anderson, D., Lunt, T., Javitz, H., Tamaru, A., Valdes, A.: Detecting unusual program behavior using the statistical components of NIDES. SRI Technical Report SRI-CRL-95-06, Computer Science Laboratory, SRI International, Menlo Park, California (1995)
11. Xu, L., Krzyzak, A., Suen, C.Y.: Methods for combining multiple classifiers and their applications to handwriting recognition. IEEE Transactions on Systems, Man, and Cybernetics **22**(3) (1992) 418–435

12. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. Second edn. John Wily & Sons, Inc., New York (2000)
13. Kittler, J., Hatef, M., Duin, R.P., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(3) (1998) 226–239
14. Kuncheva, L.: A theoretical study on six classifier fusion strategies. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(2) (2002) 281–286
15. Mazhelis, O., Puuronen, S.: Combining one-class classifiers for mobile-user substitution detection. In Seruca, I., Filipe, J., Hammoudi, S., Cordeiro, J., eds.: Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS 2004). Volume 4., Portugal, INSTICC Press (2004) 130–137
16. Raento, M., Oulasvirta, A., Petit, R., Toivonen, H.: Contextphone, a prototyping platform for context-aware mobile applications. IEEE Pervasive Computing **4**(2) (2005)
17. Oulasvirta, A., Raento, M., Tiitta, S.: Contextcontacts: Re-designing smartphone's contact book to support mobile awareness and collaboration. In: Proceedings of the 7th International Conference on Human Computer Interaction with Mobile Devices and Services, MOBILE-HCI'05, ACM (2005) 167–174
18. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann Publishers (2000)
19. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology **143** (1982) 29–36
20. Laasonen, K., Raento, M., Toivonen, H.: Adaptive on-device location recognition. In Ferscha, A., Mattern, F., eds.: PERVASIVE 2004, LNCS 3001, Springer-Verlag Berlin Heidelberg (2004) 287–304