Keystroke Analysis of Free Text

DANIELE GUNETTI and CLAUDIA PICARDI
University of Torino

Keystroke dynamics can be useful to ascertain personal identity even *after* an authentication phase has been passed, provided that we are able to deal with the typing rhythms of free text, chosen and entered by users without any specific constraint. In this paper we present a method to compare typing samples of free text that can be used to verify personal identity. We have tested our technique with a wide set of experiments on 205 individuals, obtaining a False Alarm Rate of less than 5% and an Impostor Pass Rate of less than 0.005%. Different trade-offs are, however, possible. Our approach can rely on what is typed by people because of their normal job, and a few lines of text, even collected in different working sessions, are sufficient to reach a high level of accuracy, which improves proportionally to the amount of available information: As a consequence, we argue that our method can be useful in computer security as a complementary or alternative way to user authentication and as an aid to intrusion detection.

Categories and Subject Descriptors: D.4.6 [**Operating Systems**]: Security and Protection—*Access controls, authentication*

General Terms: Experimentation, Security

Additional Key Words and Phrases: Biometric techniques, keystroke analysis of free text, identity verification

1. INTRODUCTION

Biometrics is the term used to indicate a set of physiological and behavioral human characteristics that may allow verification of personal identity. Biometric techniques, based on physiological features (such as iris and facial analysis, palm topology, hand geometry, and vein patterns), are considered more successful than those based on behavioral characteristics (such as speech, keystroke, and hand-writing analysis) [Ashbourn 2000a]. One reason is probably that physiological features are very stable and, in normal conditions, do not vary with time. On the contrary, behavioral features can be greatly influenced by transient situations, such as stress or illness. They also show a certain degree of instability even without any evident reason.

In the case of computer security, keystroke analysis is appealing for many reasons. First, it is not intrusive, since users will be typing on the computer

Authors' address: Dipartimento di Informatica, University of Torino, corso Svizzera 185, 10149 Torino, Italy; tel: +39 011 6706711, fax: +39 011 751603; email: {gunetti,picardi}@di.unito.it.

keyboard anyway. Second, it is relatively inexpensive to implement, since the only hardware required is the computer. Finally, but very important, typing rhythms are still available *after* an authentication phase has passed (or been fooled), since keystrokes exist as a mere consequence of users using computers.

A serious problem exists when analyzing typing rhythms. Keystrokes, unlike other biometric features, convey an unstructured and very small amount of information. From two consecutive keystrokes we may just extract the elapsed time between the release of the first key and the depression of the second (the so-called *digraph latency*) and the amount of time each key is held down (the *keystroke duration*), a pretty shallow kind of information. Moreover, this information may vary not only because of the intrinsic instability of behavioral characteristics, but because different keyboards can be used, different environmental conditions exist, and, above all, because typing rhythms also depend on the entered text.

In order to deal with the instability of typing rhythms, most research in the field of keystroke analysis limited the experiments to samples produced from structured, predefined text. Moreover, since having to enter long fixed texts (both when building users' profiles and during the verification phase) is obviously tedious, researchers strived to work with relatively short sample phrases. In many cases, experimental results are controversial. Either the attained level of accuracy is far from being acceptable, or good performance is achieved under very special conditions, which is hard to maintain in real applications [Bergadano et al. 2002].

We prefer to take a different approach. We believe that, at the current state of the art, keystroke analysis cannot be performed with very short texts. Timing analysis on such texts does not provide a sufficient amount of information to discriminate accurately among legal users and impostors. On the contrary, if relatively long sample texts are accepted, keystroke analysis can become a valid tool to ascertain personal identity. However, if short texts do not provide sufficient timing information, and if long predefined texts to be entered repeatedly are unacceptable, we are left with only one possible solution: using the typing rhythms users show during their normal interaction with a computer. In other words, we have to be able to deal with the keystroke dynamics of free text.

In this paper, we describe a method that analyzes typing samples of free text, which is the natural evolution of the basic technique and system [Gunetti and Bergadano 2002] we introduced in Bergadano et al. [2002], and that performed well for fixed text. We have tested our method on a wide set of experiments involving 205 volunteers, reaching a False Alarm Rate (FAR) of less than 5% and an Impostor Pass Rate (IPR) of 0.00489% (that is, less then one unnoticed impostor out of 20,000 attempts), for samples of free text of about 800 characters long, on average. A different trade-off can be easily set up in our system: for example, by accepting a FAR of 6.66%, we reach an IPR of 0.000889%: less than one unnoticed impostor out of 110,000 attacks. We suggest that the ability to deal with free text makes keystroke analysis useful within computer security, for example, as an aid to intrusion detection and as an alternative or complementary way to authenticate users, as we discuss in the last section of the paper.

2. RELATED WORK

Within keystroke analysis literature, the distinction between *static* and *dynamic* (or *continuous*) analysis is often made. Static keystroke analysis means essentially that the analysis is performed on typing samples produced using the same predetermined text for all the individuals under observation. Most systems fall within this category, [e.g., Umphress and Williams 1985; Leggett and Williams 1988; Joyce and Gupta 1990; Bleha et al. 1990; Brown and Rogers 1993; Obaidat and Sadoun 1997; Bergadano et al. 2002; Clarke et al. 2003]. The intended application of static analysis is at login time, in conjunction with, or in place of, other authentication methods.

Dynamic analysis implies a continuous or periodic monitoring of issued keystrokes and is intended to be performed *during* a log-in session, after the authentication phase has passed. Unfortunately, the term *dynamic* is often misleading. In fact, in most cases, the experiments are still performed using a unique, or a small number of predetermined texts, and the term "dynamic" accounts only for the fact that authentication is attempted before the text is complete, by using only a varying fraction of the entered keystrokes (e.g., the first 100 or 200 keystrokes), such as in Leggett et al. [1991] and Furnell et al. [1996]. However, it should be obvious that keystroke analysis performed after the authentication phase should deal with the typing rhythms of whatever is entered by the users. In other words, it should deal with *free text*. For this reason, we consider the aforementioned systems still based on essentially static keystroke analysis.

Since the terms *static* and *dynamic* may generate confusion, we prefer to speak of keystroke analysis based on *fixed* (sometimes also referred to as structured) text, or on *free* text. The former term should be adopted in all cases when the text used to perform the analysis is constrained in some way. The latter term applies when users are free to type whatever they want and keystroke analysis is performed on the available information. It should be clear that the analysis of free text is dynamic by definition, since even the length of the text is free. Of course, a minimum amount of text is needed in order to make the analysis of the typing dynamics meaningful. From this point of view, all systems cited so far should be considered as based on the analysis of fixed text.

The literature on keystroke analysis of "true" free text is pretty limited. Monrose and Rubin [1997] collected typing samples from 42 users in about 7 weeks, but experiments were only conducted on 31 users, as 11 profiles had to be eliminated due to erroneous timing results. Participants ran the experiment from their own machines at their convenience. They had to type a few sentences from a list of available phrases and/or to enter a few completely free sentences. It is unknown how many characters had to be typed by each user to form his/her own reference profile. Profiles contain the mean latency and standard deviation of digraphs as well as the mean duration and standard deviation of keystrokes. When forming a profile, each latency and duration is compared with its respective mean and any values greater than T·standard deviations above the mean (that is, outliers) are discarded from the profile. The means of the remaining

latencies and durations are then recalculated. According to the authors, for T = 0.5, more than 50% of the data was discarded.

Testing samples are manipulated in the same way by discarding outliers and so turned into testing profiles to be compared to the reference profiles using three different distance measures. A profile is treated as a N-dimensional pattern vector, where a feature (a digraph or a keystroke) has a mean value 0.0 if the number of its occurrences in the profile is less than a threshold value determined experimentally. In the first experiment, authors use the Euclidean distance between each testing and reference profile.[1] The testing profile is classified as belonging to the user whose reference profile provides the smallest Euclidean distance. In the second experiment, authors consider the actual value X_{ij} of the jth occurrence of each component X_i in a testing pattern vector U that is compared against a reference vector R. A score is computed for each X_{ij} using the mean μ_i and standard deviation σ_i of the same component in R: the higher the probability of observing the value X_{ij} in R, given μ_i and σ_i, the higher the score. U is then associated with the person who provided the reference vector that maximizes the score. The last experiment is made in a similar way, but adding weights to digraphs, in order to give more importance to those that occur more frequently in English (such as *er* or *th* w.r.t. *qu* or *ts*). When both the samples to be classified and users' profiles are of fixed text, the authors attain an outcome of about 90% of correct classification. However, when free text is used both to build users' profiles and for the samples to be classified, performance falls to 23% of correct classification in the best case (that is, using the weighted probability measure).

Works reported in Dowland et al. [2001] and Dowland et al. [2002] are interesting, especially for the adopted experimental setting. In Dowland et al. [2001] four users are monitored continuously for some weeks during their normal activity on computers running Windows NT. Hence, the production of typing samples by the users is not constrained in any way. Users' profiles are determined by computing the mean and standard deviation of digraph latency; only digraphs that occur a minimum number of times in different typing samples are kept. Nonetheless, profiles contain thousands of digraphs. A new sample U to be classified is compared against users' profiles, so that each digraph D in U is marked as "accepted" by the system if its mean latency falls within the interval defined by $\mathrm{D}_\theta^p \pm w \cdot \mathrm{D}_\sigma^p$ (where D_θ^p and D_σ^p are the mean and standard deviation of the corresponding digraph in users' profiles and w is a weighting factor). U is then attributed to the user whose profile provides the largest number of "accepted" digraphs. In this way, 50% of correct classification is achieved. The experiments are refined in Dowland et al. [2002], which includes five users and reaches correct acceptance rates slightly below 60%. No global information is given about the rejection rate of a user's sample compared against someone else's profile, but, in the case of two users, it looks to be about 75% for one user and about 85% for the other.

[1]That is, the Euclidean distance between the two corresponding vectors of mean values.

To our knowledge, the above works are the only ones concerning keystroke analysis of free text found in the literature. Thus, we are left to analyze the aforementioned systems performing "dynamic" keystroke analysis. We refer to Bergadano et al. [2002] for a description of "static" keystroke analysis systems. In Leggett et al. [1991], 36 individuals were asked to type the same text of 537 characters twice, with a delay of at least 1 month between the first and the second sample. The first sample is used as a model of the user, while the second sample is analyzed dynamically to be accepted or rejected. The second sample of each individual is also used to "attack" every other individual. As a consequence, there are 36 legal connection attempts and 1260 attack attempts (each individual pretends to be one of the other 35 ones). To build a user's model, authors extract from the first sample the eight digraphs that occur more frequently and compute their mean latency and relative standard deviation. The second sample of each user is compared to a user's profile as follows. Starting with the first digraph of the sample, the algorithm analyzes the next digraph in the sample and takes one of three actions: (1) accepts the sample as belonging to the same user of the reference profile; (2) rejects the sample as belonging to an impostor, and (3) neither accepts nor rejects the sample, but continues with the next digraph in the sample.

In order to make a decision, at every incoming digraph D, the algorithm behaves essentially like in the previous described work. If D does not occur in the current user's profile, the algorithm simply moves to the next digraph in the sample. Otherwise, the mean latency of D, D_θ, is compared with the corresponding mean (D_θ^p) and standard deviation (D_σ^p) in the profile. D is "accepted" if D_θ falls within $\mathrm{D}_\theta^p \pm \delta \cdot \mathrm{D}_\sigma^p$ for some positive value δ, to be determined experimentally. Otherwise, D is "rejected." The algorithm collects a number of "accepted" and "rejected" digraphs as it moves from one to the next in the sample. When a certain ratio A/R of "accepted" versus "rejected" digraphs is reached, the algorithm can finally accept or reject the entire sample. By careful tuning of parameters δ and A/R, on the basis of the available samples, the authors are able to reach a 11.1% FAR and a 12.8% IPR over the whole text, with many of the impostors rejected within the first 100 keystrokes of the testing sample.

In Furnell et al. [1996], 30 subjects were required to enter the same reference text of 2200 characters twice. These samples are used to form users' profiles by storing latency mean and standard deviation of digraphs. Each user was also required to provide other two samples of 574 and 389 characters that were used to attack all other users, for a total of $29 \cdot 30 \cdot 2 = 1740$ impostors' attacks. Much like in the case of the last two works described, a digraph D in the attacking sample is marked as "invalid" by the system if its latency does not fall within the interval defined by $\mathrm{D}_\theta^p \pm 1.5 \cdot \mathrm{D}_\sigma^p$, where P is the profile of the attacked user. If the ratio between the number of invalid digraphs and the total number of digraphs in the attacking sample reaches a predetermined threshold, the sample is rejected. For each user, the typing samples of 574 and 389 characters are used to set up a *personal threshold* in order to guarantee that the two samples would be accepted if compared against the reference profile of the user. In other words, the system is set up in advance to have a 0% FAR and then run to see what happens to the corresponding IPR. Authors report a 15% IPR with

26% of the impostors detected when only the first 40 characters of the testing samples are used; most of the impostors are detected within 160 keystrokes.

We also experimented within "dynamic" keystroke analysis. In Bergadano et al. [2003], we asked 40 people to enter from 2 to 5 times the same two different texts each one about 300 characters long. Adopting essentially the same experimental setting and technique described in Bergadano et al. [2002], we reach an identification accuracy of slightly more than 95%. In Bergadano et al. [2003], we show experimentally that keystroke analysis of free text, although more difficult than using fixed text, is possible: on average, two samples of different texts provided by the same individual are more similar than two samples of the same text provided by different individuals.

3. MEASURING THE DISTANCE BETWEEN TWO TYPING SAMPLES

A typing sample is simply text entered at a computer keyboard, together with some timing information about the keystrokes that produced the sample. Timing information is normally represented by two basic measures: the time a key is depressed and the time the key is released. Such information is used to compute the *duration* of a keystroke and the *latency* between two consecutive keystrokes. In our experiments, the only information sampled is the time at which a key is depressed. This allows the computation of what we call the *duration* of an n-graph: the elapsed time between the depression of the first and of the nth (that is, the last) key of a sequence of typed keys. The duration of an n-graph is clearly a combination of the latency between keystrokes and of their durations.

A typing sample can be represented in terms of the n-graphs it is composed of, together with the duration of each n-graph. If the typed text is sufficiently long, the same n-graph may occur more than once. In such cases, the n-graph is reported only once and the mean duration of its occurrences is used. It is clear that the same typing sample can be represented in terms of its digraphs (i.e., two-graphs), trigraphs (or three-graphs), four-graphs, and so on.

Given two typing samples, we want to compare them regardless of the typed text. Thus, we must extract the information they share: the n-graphs occurring in both samples. Some *distance measure* can then be applied to the extracted information. By extracting the n-graphs shared between two samples, we are automatically able to cope with typing errors and, in principle, we could also compare samples made using different languages, provided the two languages share some legal n-graph.

As an example that will be used in the remainder of this section to illustrate how we compute the distance between two typing samples, suppose that two samples **E1** and **E2** have been produced entering, respectively, the texts *authentication* and *theoretical*. A possible outcome of the samplings may be the following, where the number before each letter represents the hypothetical time in milliseconds at which the corresponding key was depressed:

E1: 0 **a** 180 **u** 440 **t** 670 **h** 890 **e** 1140 **n** 1260 **t** 1480 **i** 1630 **c** 1910 **a** 2010 **t** 2320 **i** 2600 **o** 2850 **n**

E2: 0 **t** 150 **h** 340 **e** 550 **o** 670 **r** 990 **e** 1230 **t** 1550 **i** 1770 **c** 1970 **a** 2100 **l**

In the next subsections, we will use **E1** and **E2** to describe two classes of distance measures used to perform the experiments presented in the paper. Given any two typing samples, only the n-graphs they have in common will be used to compute their distance w.r.t the measures we describe below. We will see that the two classes of measures are in some way orthogonal and try to point out different aspects of the differences and similarities between the samples under comparison. The basic measures we will describe return a real value between 0 and 1 and, for reasons explained below, they will be called "R" (standing for "Relative") and "A" (for "Absolute"). Different R and A measures will then be combined together, in this way increasing their discriminating power.

3.1 "R" Measures

The basic idea behind measure R was introduced in Bergadano et al. [2002], where many experimental properties of the measure were shown. Here, we first review the main points and then generalize the measure to a class of measures.

Given a array V of K elements, a simple measure of the *degree of disorder* (or, simply, the *disorder*) of V w.r.t. its ordered counterpart V′ can be computed as the sum of the distances between the position of each element in V and the position of the same element in V′. As an example, it easy to see that the disorder of array A = [2,5,1,4,3] w.r.t. A′ = [1,2,3,4,5] is: $(1 + 3 + 2 + 0 + 2) = 8$.

Clearly, a sorted array V′ has a disorder equal to 0, while we have the maximum disorder for an array V″ when its elements are in reverse order. Such value of maximum disorder is given by:

$$|V''|^2/2 \text{ (if } |V''| \text{ is even)}; \qquad (|V''|^2 - 1)/2 \text{ (if } |V''| \text{ is odd)}$$

Given an array of K elements, it is convenient to normalize its disorder by dividing it by the value of the maximum disorder of an array of K elements. In this way it is possible to compare the disorder of arrays of different size. After this normalization, the disorder of any array V falls between 0 (if V is ordered) and 1 (if V is in reverse order). The normalized disorder of array A above is: $(1 + 3 + 2 + 0 + 2)/[(5^2 - 1)/2] = 8/12 = 0.66666$.

Now, let us consider two typing samples **S1** and **S2** and, for a given n, let them be represented as two arrays sorted w.r.t. the duration of their n-graphs. We may consider one of the two (e.g., **S1**) as the referring sorted array and we may compute the *distance* between **S1** and **S2** w.r.t. the n-graphs they share (in short, R_n(**S1**, **S2**)) as the normalized disorder of **S2** w.r.t. **S1**. In other words, if **S1** and **S2** share K n-graphs, the distance of **S2** from **S1** is the sum of the distances of each n-graph of **S2** w.r.t. the position of the same n-graph in **S1**, divided by the maximum disorder showed by an array of K elements; n-graphs not shared between the two samples are simply ignored.

It is clear that R_n(**S1**, **S2**) = R_n(**S2**, **S1**) and it also clear that R_n can be used to compute the distance between any two typing samples, provided they share some n-graph (which is normally the case for texts sufficiently long—e.g., a full line of text—and for n sufficiently small—e.g., $n < 5$). For example, consider

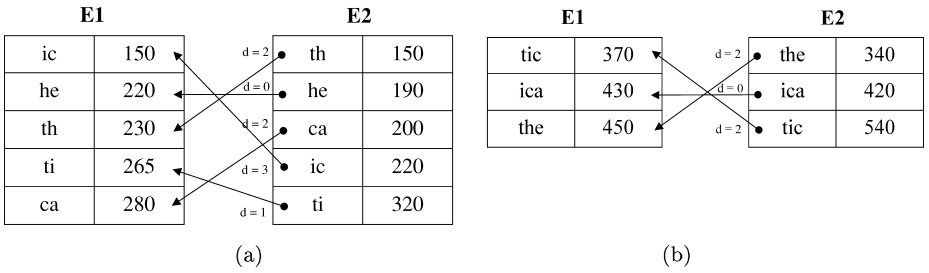

Fig. 1. (a) Computation of the distance of two typing samples using digraphs. (b) Computation of the distance of two typing samples using trigraphs.

the two typing samples **E1** and **E2** described above. To compute R_2(**E1**, **E2**), we first extract the digraphs shared by **E1** and **E2** and then we may compute R_2(**E1**, **E2**), as depicted in Figure 1(a).[2] The two samples share five digraphs and the maximum disorder for an array of five elements is $(5^2 - 1)/2 = 12$, so that we have:

$$R_2(\mathbf{E1}, \mathbf{E2}) = (2 + 0 + 2 + 3 + 1)/12 = 8/12 = 0.6666$$

Similarly, **E1** and **E2** share three trigraphs, so that (see Figure 1(b)) we have:

$$R_3(\mathbf{E1}, \mathbf{E2}) = (2 + 0 + 2)/[(3^2 - 1)/2] = 1$$

In the case of samples **E1** and **E2**, it is not possible to compute R_n for $n > 3$, since the two samples do not share any four-graph. Clearly, longer samples would allow the computation of R_4, R_5, and so on.

Intuitively, the meaningfulness of a distance R_n between two samples **S1** and **S2** is related to the number of n-graphs shared by the two samples (we showed this experimentally in Bergadano et al. [2002] for the case R_3, which was the measure used therein). Hence, in the case of our example, we may consider R_2(**E1**, **E2**) more meaningful than R_3(**E1**, **E2**), since the former is computed using five digraphs and the latter using only three trigraphs (though for such short texts, both R_2 and R_3 are relatively meaningless).

It is reasonable to expect that the computation of the relative distance between two typing samples, w.r.t. the digraphs, trigraphs, or longer n-graphs they share, may provide different kinds of information about the similarities and differences between the two samples. This is particularly true if the samples have been produced entering different texts. For example, the typing speed of a certain digraph occurring in both samples may be influenced by the fact that the digraph appears in different contexts (that is, as part of different trigraphs) in the two samples.

Hence, a more meaningful measure of the distance between two samples **S1** and **S2** may be obtained by combining R_n(**S1**, **S2**) for different instances of n. When doing so, we must weight each instance of R_n, and the most obvious way

[2]In the figure, next to every digraph is its duration in milliseconds, that can be computed from the hypothetical times reported in the page 317. Note that digraph *ti* occurs twice in **E1**, so we use the mean of the two durations.

to do so is on the basis of the number of n-graphs used to compute the corresponding R_n, so that more n-graphs account for a more accurate R_n. Formally, if two samples **S1** and **S2** share N n-graphs and M m-graphs, with N > M, we may define:

$$R_{n,m}(\mathbf{S1}, \mathbf{S2}) = R_n(\mathbf{S1}, \mathbf{S2}) + R_m(\mathbf{S1}, \mathbf{S2}) \cdot M/N$$

In other words, the *cumulative* distance $R_{n,m}$ between two samples is the sum of the two distances R_n and R_m weighted by the different number of n-graphs and m-graphs they share. For example, in the case of samples **E1** and **E2** we have:

$$R_{2,3}(\mathbf{E1}, \mathbf{E2}) = 0.6666 + 1 \cdot 3/5 = 1.2666$$

Of course, the cumulative R distance between two samples may be extended to a larger number of different graphs. That is:

$$R_{n,m,p}(\mathbf{S1}, \mathbf{S2}) = R_n(\mathbf{S1}, \mathbf{S2}) + R_m(\mathbf{S1}, \mathbf{S2}) \cdot M/N + R_p(\mathbf{S1}, \mathbf{S2}) \cdot P/N$$

where P is the number of p-graphs shared by **S1** and **S2** and N > M, N > P. Clearly, even more different graphs may be taken into consideration, but when the weighting ratio P/N becomes very small, the contribution of the p-graphs shared by **S1** and **S2** is negligible. In practice, for samples no longer than a few hundreds characters, it is probably worth taking into consideration at most digraphs, trigraphs, and four-graphs (see also the observation at the end of Section 5.2).

Readers may have noticed that measures R completely overlook any absolute value of the timing features of the samples under comparison. For example, even if the typing speed of all digraphs in sample **E1** were twice the values reported, this would not be captured in any way by measure R, and R_2(**E1**, **E2**) would still be 0.666: only the relative typing speed of digraphs in the two samples are taken into consideration—hence, the name "R."

The rationale behind R measures is that the typing speed of an individual may change along with the psychological and physiological conditions of the subject, but we may expect the changes to affect all the typing characteristics in a similar way. A headache can cause the individual to type more slowly than usual, but the relative typing speed of the entered n-graphs will probably remain stable. If the individual normally types the four-graph *wait* more slowly than the four-graph *stop*, this is likely to remain unchanged, even with an headache.

R_n measures are simple metrics defined over $\mathcal{S}_n^M \times \mathcal{S}_n^M \rightarrow \Re$, where $\mathcal{S}_n^M$ is the space of all typing samples sharing M different n-graphs, sorted w.r.t. their typing speed. Clearly, $\mathcal{S}_n^M$ is finite for a given set of M n-graphs, and contains $M!$ elements.[3] If **S1**, **S2**, and **S3** belong to $\mathcal{S}_n^M$ for $n \geq 2$ and $M > 0$, from the definition of R_n it is easy to see that $R_n(\mathbf{S1}, \mathbf{S2}) \geq 0$ (and $R_n(\mathbf{S1}, \mathbf{S2}) = 0$ if, and only if, $\mathbf{S1} = \mathbf{S2}$) and $R_n(\mathbf{S1}, \mathbf{S2}) = R_n(\mathbf{S2}, \mathbf{S1})$. To prove the triangular inequality $R_n(\mathbf{S1}, \mathbf{S3}) \leq R_n(\mathbf{S1}, \mathbf{S2}) + R_n(\mathbf{S2}, \mathbf{S3})$, we observe, first of all, that the inequality is trivially true if two of the three samples coincide. Otherwise, note that

[3]Assuming all n-graphs have a different typing speed.

R_n(**S1**, **S3**) is the minimum number of positions that all n-graphs in **S3** must shift to reach the position they have in **S1** (apart from the normalization factor, which is, of course, the same for all pair of samples in $\mathcal{S}_n^M$). But then R_n(**S1**, **S3**) > R_n(**S1**, **S2**) + R_n(**S2**, **S3**) is impossible, since it would mean that R_n(**S1**, **S3**) is not the minimum number of shifts needed to turn **S3** into **S1**, and it would be possible to get **S1** from **S3** "passing" through **S2** in only R_n(**S1**, **S2**) + R_n(**S2**, **S3**) shifts.

3.2 "A" Measures

The distance measure R_3 was introduced in Bergadano et al. [2002] and in the case of fixed text provided good results. Unfortunately, R measures may also show some counterintuitive behavior. Consider, for example, the case where the typing speed of each digraph in a sample **S1** is exactly twice the typing speed of the same digraph in a sample **S2**. In this case we have R_2(**S1**, **S2**) = 0. Hence, R fails to discriminate between the typing samples of two typists that have very similar typing rhythms, even if one of the typists is much faster than the the other one. As a consequence, we need to be able to grasp similarities and differences between typed samples also on the basis of the absolute typing speed they are entered.

Unlike R measures, A measures only consider the absolute value of the typing speed of each pair of identical n-graphs in the two samples under comparison. The rationale behind measures A is simple. Suppose a user is able to type on a keyboard exactly in the same way at all times. Then, a particular n-graph is always typed in the same way and, in particular, always at the same speed regardless of the entered text. Of course, this is practically impossible, but we may reasonably expect two occurrences of a given n-graph typed by the same individual, even in two different texts, to have durations that are closer than if the two texts have been entered by different individuals.

More formally, let $G_{S1,d1}$ and $G_{S2,d2}$ be the same n-graph occurring in two typing samples **S1** and **S2**, with durations d1 and d2, respectively. We say that $G_{S1,d1}$ and $G_{S2,d2}$ are *similar* if $1 < \max(d1, d2)/\min(d1, d2) \leq t$ for some constant t greater than 1. The "A" distance between **S1** and **S2** w.r.t. the n-graphs they share and for a certain value of t is then defined as:

A_n^t(**S1**, **S2**) = 1 − (number of similar n-graphs between **S1** and **S2**)/(total number of n-graphs shared by **S1** and **S2**)

As a consequence, if there are no similar pairs of n-graphs in two typing samples, their distance is 1. If all n-graphs pairs are similar, the distance is 0. From the above definition of distance, we see that two typing samples that share no n-graph have 1 as the maximum distance possible. On the contrary, the two samples will have the minimum distance w.r.t. any R measure. Actually, we do not mind such extreme situations, since they are virtually impossible for samples sufficiently long.

Although A measures may slightly resemble the Hamming distance, there are differences. A measures count similar n-graphs instead of nonsimilar ones

and a threshold t is used to establish when corresponding n-graphs are *similar*. Moreover, A distances are normalized in the interval [0–1] to handle pairs of samples sharing a different number of n-graphs. Finally, unlike the Hamming distance, A measures are not true metrics, since the triangular inequality is not satisfied.[4]

The rule used to mark n-digraphs as *similar* is pretty close to the way digraphs are marked as "accepted" or "invalid" in other approaches to keystroke analysis, as we saw in Section 2. There is, however, an important difference. We do not use standard deviations to define an interval of validity for n-graphs, but prefer to rely on a fixed ratio (t) between the typing speeds of n-graphs. In this way, even n-graphs occurring only once in both samples (so that for such n-graphs a true mean speed or standard deviation cannot be computed) can be compared. This helps, of course, to exploit as much as possible all the available timing information, even in samples that may not have n-graphs repeated many times (which is more and more likely as the length of graphs increases).

In the case of samples **E1** and **E2** above, for digraphs and for $t = 1.25$ we have:

E1		**E2**		
280	**ca**	200	$(280/200 = 1.400)$	
220	**he**	190	$(220/190 = 1.157)$	(similar pair)
150	**ic**	220	$(220/150 = 1.466)$	
230	**th**	150	$(230/150 = 1.533)$	
265	**ti**	320	$(320/265 = 1.207)$	(similar pair)

And $\mathrm{A}_2^{1.25}(\mathbf{E1}, \mathbf{E2}) = 1 - 2/5 = 0.6$. It is easy to check that $\mathrm{A}_3^{1.25}(\mathbf{E1}, \mathbf{E2}) = 2/3$.

Much like we saw for measures R, for a given t, it is possible to compute a cumulative absolute distance between two typing samples w.r.t. n-graphs of different length:

$$\mathrm{A}_{n,m}^{t}(\mathbf{S1}, \mathbf{S2}) = \mathrm{A}_{n}^{t}(\mathbf{S1}, \mathbf{S2}) + \mathrm{A}_{m}^{t}(\mathbf{S1}, \mathbf{S2}) \cdot \mathrm{M/N}$$

for M m-graphs and N n-graphs shared by **S1** and **S2** and $\mathrm{N} > \mathrm{M}$.[5] In a similar way, we may define $\mathrm{A}_{n,m,p}^{t}(\mathbf{S1}, \mathbf{S2})$.

In order for A measures to become operative, a suitable value for t must be chosen. In fact, a serious drawback that may affect systems based on experimental data is the well-known problem of *overfitting*: a tailoring of a method on a particular set of data in order to achieve the best results. Such tailoring would very probably fail to achieve the same results with a different set of samples. In the case of our experiments, overfitting may occur when choosing a value for t.

To limit the problem, we did the following. When the first five volunteers of our experiments had provided their samples, we performed the classification task described in Section 5.1, in order to test measure A_2^t for different values of t. We experimented with values 1.20, 1.25, 1.33, and 1.40. The best outcomes were reached for $t = 1.25$ and, hence, this value is the one used for measures A

[4]As a simple example, consider three samples containing only digraph ab with durations: $\mathbf{S1} = 100$ ms, $\mathbf{S2} = 115$ ms, $\mathbf{S3} = 130$ ms. Then, $1 = \mathrm{A}_2^{1.25}(\mathbf{S1}, \mathbf{S3}) > \mathrm{A}_2^{1.25}(\mathbf{S1}, \mathbf{S2}) + \mathrm{A}_2^{1.25}(\mathbf{S2}, \mathbf{S3}) = 0$.

[5]One might also want to weight n-graphs w.r.t. the number of their occurrences in a sample, but in our experiments this did not provide any improvement to the performance of A measures.

in all of the experiments reported in this paper, for digraphs as well as for longer n-graphs. Actually, it is possible that better outcomes could be reached for some value in the interval [1.20,1.40], and that different values may perform better for trigraphs and four-graphs, but we did not bother to find such particular values that would hardly perform as well on a different set of users. Since $t = 1.25$ is the value used in all the experiments described below, in the rest of the paper we will omit it as the superscript of A measures.

4. EXPERIMENTAL SETTING

All the typing samples used to test the performance of measures R and A were collected through a simple HTML form comprised of two fields. In the first field, users had to enter their login name. The second field was a text area where volunteers were free to enter whatever they wanted to fill the form. When the form was complete, a "submit" button was used to send the sample to the collecting server. A client-side Javascript was written to gather the timing information. The sampling data was composed of the time (in milliseconds) when a key was depressed, together with the ascii value of the key. The timer had a clock tick of 10 ms.

4.1 The Volunteers and the Gathering of Samples

Fourty volunteers provided 15 typing samples each, as described below. In our experiments, these people act as legal users of a hypothetical system. Moreover, under the same conditions described below, another 165 people were asked to provide just one typing sample. These individuals act as impostors that pretend to be one of the legal users of the system.

All the people participating in the experiments were native speakers of Italian and were asked to provide samples written in Italian. We found the volunteers in our department, among colleagues and students in their last year of study. Although with varying typing skills, all of them were very used to typing on normal computer keyboards. Of course, none of the volunteers was hired, or in any way paid for their assistance.

The samples were collected on the basis of the availability and willingness of volunteers over a period of about 6 months. All volunteers were instructed to provide no more than one sample per day. A few individuals were able to provide their samples on a very regular basis, each working day or every 2 or 3 days, so they completed their task in about 1 month. However, the majority of the participants to the experiment provided their typing samples on a very irregular basis; some of the samples were provided frequently, while other samples every 2 or 3 weeks. For others still, even 1 or 2 months passed between the production of a sample and the next.[6] Samples were provided at any working hour, at the convenience of the participants.

For obvious reasons, we had no control over the way volunteers performed the experiment. Every working day an email was sent to each of the volunteers to remind them about the experiment, but people were, of course, free

[6]Such large intervals were due to different reasons, including vacations.

to ignore such reminders. Most of the volunteers used their office computers or their notebook computers to produce the samples. In some cases, the experiment was done by some volunteer connecting from home or from computers located in another town. Thus, each user may have provided some of his/her samples using different keyboards. However, we do not have any way to know which samples were provided on which keyboards, and we have not explicitly tested our method w.r.t. the use of different keyboards.[7] Samples were provided both in Windows and Unix environments, using both Explorer and Netscape browsers.

Volunteers were asked to enter the samples in the most natural way, more or less as if they were writing an email to someone. They were completely free to choose what to write and the only limitations were of not typing the same word or phrase repeatedly in order to complete the form and not to enter the same text in two different samples. We suggested that volunteers write about different subjects in each sample: their job, movies, holidays, recipes, family, and so on—anything they liked. They were, of course, free to make typos and to correct them or not. Corrections could be made by using the backspace key or the mouse, as preferred. People were free to pause for whatever reason and for as long as they wanted when producing a sample. No sample provided by the volunteers was rejected, for any reason.

The text area to be filled was 65 characters wide and 12 lines long, for a total of 780 characters. However, volunteers were not compelled to enter exactly 780 characters, but just to stop when they had the "feeling" the form had been filled. On average, the gathered samples have a length varying between 700 and 900 characters (just to give an idea, the previous paragraph is about 850 characters long).

5. USER CLASSIFICATION, AUTHENTICATION, AND IDENTIFICATION

Within Biometrics, the distinction between user authentication and identification is often made. In the first case, a typing sample is provided together with a declaration of identity. The system must decide if the sample comes from the user whose identity has been declared or not. In user identification the system only sees a new sample and must determine who provided the sample, or that the sample does not come from any of the users known to the system. Clearly, the task of user identification is more difficult than user authentication [Ashbourn 2000b]. However, this is true only if new samples presented to the system may also come from individuals different from system's users, individuals whose

[7]It would also be interesting to test our method with other kinds of input devices, such as, e.g., terminal for the blinds (although blind people also use standard keyboards, sometimes equipped with Braille codes on the keys). Such terminals have from six to eight large keys that must be pressed to form the different combinations of dots of the Braille code. Additional control keys are often available. Typing styles would clearly be quite different from than on QWERTY keyboards. However, as long as a sufficient amount of timing data can be gathered, the keystroke analysis would still be possible. On the contrary, our method would not be suitable to analyze the very short key sequences normally entered on numerical keypads, such as phone numbers or pins. Also raw timing data stemming from mouse movements and clicks would probably not fit well in our framework, which requires a sufficient number of different typing patterns—the n-graphs durations—to work.

typing habits are completely unknown to the system. If new samples only come from individuals known to the system, both authentication and identification become much easier, but system performance is less meaningful: After all, in many real security applications, most of the attacks are to be expected from the outside world, by someone who is unknown to the system.

Unfortunately, the above is not always well understood in the literature concerning keystroke analysis. In some cases, systems claim to perform identification, but when they are tested, some knowledge is provided about the typing habits of the attackers, a situation that is closer to classification than to identification [for example, this is the case for Leggett et al. 1991, Brown and Rogers 1993, and Furnell et al. 1996]. On the contrary, Leggett and Williams [1988], Leggett et al. [1991], Joyce and Gupta [1990], and Bergadano et al. [2002] perform true authentication/identification, whereas this is unclear in the experiments described in Bleha et al. [1990].

Hence, a system performing keystroke analysis can be tested on three tasks of increasing difficulty, when a new sample X is submitted to the system:

- **Classification:** X comes from one of the known users. The system must find who actually provided the sample.
- **Authentication:** X is claimed to belong to user U. The system must decide if this is true or false. X may belong to U, to another known user, or to someone else (whose typing habits are) completely unknown to the system.
- **Identification:** X is presented to the system. The system has two possible answers: (a) X belongs to user U; or (b) X belongs to someone unknown. As in the case of authentication, X may, in fact, belong to one of the known users, or to someone unknown to the system.

In the following subsections we will cope with the above three tasks using the distance measures introduced earlier, building more sophisticated methods on top of the more simple ones.

5.1 User Classification

To perform user classification, we start with the same technique we introduced in Bergadano et al. [2002]. However, here we will use different ways to compute the distance between two samples, so as to individuate those that perform better.

Let d be a generic distance measure between two samples **S1** and **S2**, such that d(**S1**, **S2**) ≥ 0. Let A be a user and let the typing profile of A be made of a set {A1, A2, . . . , An} of typing samples provided by A. We define the *mean distance* (*md* for short) of a typing sample X from A's profile as:

$$\mathrm{md(A,X)} = [\mathrm{d(A1, X)} + \mathrm{d(A2, X)} + \cdots + \mathrm{d(An, X)}]/\mathrm{n}$$

Hence, if there is a set of users A, B, C, . . . , and a typing sample X to be classified as belonging to one of the users, we may classify X as coming from the user with the smallest mean distance from X.

Our experiments in user classification are made using the samples gathered from the 40 users each one providing 15 samples. Each user's profile

Table I. Experimental Results in User Classification for Different R and A Measures[a]

a. R Measures							
Adopted distance measure	R_2	R_3	R_4	$R_{2,3}$	$R_{2,4}$	$R_{3,4}$	$R_{2,3,4}$
No. of classification errors	13	44	61	5	9	29	9
% of error	2.16	7.33	10.16	0.83	1.5	4.83	1.5
b. A Measures							
Adopted distance measure	A_2	A_3	A_4	$A_{2,3}$	$A_{2,4}$	$A_{3,4}$	$A_{2,3,4}$
No. of classification errors	44	84	133	41	39	81	41
% of error	7.33	14.0	22.16	6.83	6.5	13.5	6.83

[a]Attempted classifications: 600.

contains 14 samples and the remaining one must be classified. Hence, by using every sample as a test sample, we have a total of $15 \cdot 40 = 600$ attempted classifications.

Table Ia illustrates the outcomes obtained by using, as distance measure, all R distances that may be computed on the samples using digraphs, trigraphs, and four-graphs. Table Ib shows the same experimental outcomes using A distances.

From the two tables it is easy to see that (1) R measures perform better than the corresponding A measures; (2) distances computed using graphs of different lengths perform better than the same distances used separately (this is especially evident for R measures. For example, $R_{2,3}$ provides better results than R_2 and R_3); (3) on average, distance measures involving shorter graphs perform better than measures involving longer graphs. For example, R measures including digraphs always perform better than R measures not using digraphs, and R measures involving trigraphs are more accurate than R measures using four-graphs in place of trigraphs. In the case of A measures, this is less evident: A measures involving digraphs are more accurate than A measures using only longer graphs, but adding trigraphs or four-graphs, or both, to digraphs seems to provide more or less the same improvement.

As a further step, let us see what happens when R and A measures are summed together to compute the distance between two samples. For example, in the following, when we write that the adopted distance measure is $R_2 + A_{2,3}$ we mean that the distance between any two samples **S1** and **S2** is computed as: $d(\mathbf{S1}, \mathbf{S2}) = R_2(\mathbf{S1}, \mathbf{S2}) + A_{2,3}(\mathbf{S1}, \mathbf{S2})$. For the sake of conciseness, we only combine in all possible ways the four R measures that provide the best results (R_2, $R_{2,3}$, $R_{2,4}$, $R_{2,3,4}$) together with the four A measures that provide the best results (A_2, $A_{2,3}$, $A_{2,4}$, $A_{2,3,4}$). Outcomes are shown in Table II (distance measures computed using $R_{2,4}$ provide outcomes quite similar to those reached using $R_{2,3}$, so we omit them for brevity).

From Table II it is easy to notice a further improvement in the outcomes due to the combination of R and A measures. One may observe that such improvement is pretty small, since only a few more samples are correctly classified. Actually, R measures provide an accuracy pretty close to 100%, so that a little space for improvements is left. As we will see in the next section, the positive influence of combining R and A measures together is much more evident within authentication.

Table II. Experimental Results in User Classification for Different R and A Measures Combined[a]

a				
Adopted distance measure	$R_2 + A_2$	$R_2 + A_{2,3}$	$R_2 + A_{2,4}$	$R_2 + A_{2,3,4}$
No. of classification errors	6	4	11	14
% of error	1.0	0.66	1.83	2.33
b				
Adopted distance measure	$R_{2,3} + A_2$	$R_{2,3} + A_{2,3}$	$R_{2,3} + A_{2,4}$	$R_{2,3} + A_{2,3,4}$
No. of classification errors	2	4	5	7
% of error	0.33	0.665	0.83	1.16
c				
Adopted distance measure	$R_{2,3,4} + A_2$	$R_{2,3,4} + A_{2,3}$	$R_{2,3,4} + A_{2,4}$	$R_{2,3,4} + A_{2,3,4}$
No. of classification errors	2	1	4	4
% of error	0.33	0.16	0.66	0.66

[a]Attempted classifications: 600.

5.2 User Authentication

Suppose that we have three users A, B and C, and let X be a new sample to be classified, so that we compute, for a distance measure d:

$$\mathrm{md(A, X)} = 0.419025;\ \mathrm{md(B, X)} = 0.420123;\ \mathrm{md(C, X)} = 0.423223$$

Hence, we decide that X belongs to user A, following the classification rule of the previous section. However, for a given user A, let m(A) be the mean of the distances of the samples in A's profile. In fact, m(A) can be seen as a sort of number representative of the way A types on a keyboard: m(A) gives us an idea of the distance we may expect between two typing samples provided by user A. Now, suppose that in our example A's samples are A1, A2, and A3, and that we have:

$$\mathrm{d(A1,A2)} = 0.312378;\ \mathrm{d(A1,A3)} = 0.304381;\ \mathrm{d(A2,A3)} = 0.326024;$$
$$\mathrm{m(A)} = (0.312378 + 0.304381 + 0.326024)/3 = 0.314261$$

Then, we may expect another sample of A to have a mean distance from the model of A similar to m(A), which is not the case for X in the example above. Even if X is closer to A than to any other user's profile in the system, it should be rejected.

In other words, the classification rule described in the previous section cannot be used to perform authentication. An impostor pretending to be a legal user U of a system, has 1/N chances of being recognized as U, if there are N legal users in the system. This happens whenever the impostor's sample is closer to the profile of the user under attack than any other legal user. Hence, from the above rule we may expect, at best, a (100/N)% IPR.

However, as observed in Bergadano et al. [2002], we can do much better, by refining the classification rule as follows: a sample X claimed to belong to user A is acknowledged as coming from A if, and only if,

(1) md(A,X) is the smallest w.r.t. any other md(B,X), where B is another legal user in the system;
(2) md(A,X) is smaller than m(A) (2a), **or** md(A,X) is closer to m(A) than to any other md(B,X) computed by the system (2b).

Requirement (1) is the basic classification rule: X may come from A only if X is *closer* to A's samples than to the samples of any other user B. However, we also want to check whether X is *sufficiently close* to A in order to accept it. This is what requirement (2) states. As we saw, m(A) is the expected distance between samples from A. If (requirement 2a) X is so similar to A's samples that md(A,X) is even smaller than m(A), X is recognized as belonging to A. If (requirement 2b) md(A, X) is greater than m(A), then we check if the distance between X and A's samples is closer to the distance we expect from A's samples, m(A), than to the distance that X shows w.r.t the samples of any other user B. Since 2b is asked only when md(A,X) > m(A) and, moreover, md(A,X) < md(B,X), we can express 2b formally as follows:

$$\begin{aligned}
&\mathrm{md(A, X)} - \mathrm{m(A)} < \mathrm{md(B,X)} - \mathrm{md(A, X)};\\
&\mathrm{md(A, X)} - \mathrm{m(A)} < \mathrm{m(A)} - \mathrm{md(A, X)} + \mathrm{md(B, X)} - \mathrm{m(A)};\\
&2\mathrm{md(A, X)} < 2\mathrm{m(A)} + \mathrm{md(B,X)} - \mathrm{m(A)};\\
&\mathrm{md(A, X)} < \mathrm{m(A)} + 0.5[\mathrm{md(B,X)} - \mathrm{m(A)}]
\end{aligned}$$

Note that this inequality is trivially true if md(A,X) < m(A); thus, it actually expresses the whole of requirement (2).

One may wonder why not use a more intuitive rule, such as, $|\mathrm{md(A,X)} - \mathrm{m(A)}| < |\mathrm{md(B,X)} - \mathrm{m(B)}|$ in place of requirement (2). However, there are two problems with this inequality.

First, suppose requirement (1) is passed (that is, md(A,X) < md(B,X)) by a lucky impostor pretending to be A. Then, recalling the example at the beginning of this section, it is very likely that md(A,X) $\simeq$ md(B,X). Using the rule $|\mathrm{md(A,X)} - \mathrm{m(A)}| < |\mathrm{md(B,X)} - \mathrm{m(B)}|$, X will be accepted or rejected depending on whether m(A) is greater or smaller than m(B), that is, by chance. On the contrary, with rule 2b, when md(A,X) $\simeq$ md(B,X) X is accepted only if md(A,X) and m(A) are even closer. In other words the tightness of requirement 2b is proportional to how well X passed requirement (1).

Second, suppose B provided samples with a very high variability between each other, so that m(B) is high (that is, m(B) $\gg$ m(A) for any other A). Then, if X belongs to A, md(B,X) will also be high and there are good chances that $|\mathrm{md(A,X)} - \mathrm{m(A)}| < |\mathrm{md(B,X)} - \mathrm{m(B)}|$ does not hold. Even if md(A,X) < md(B,X), this is blurred by the high variability of the typing habits of B. Thus, the presence of a user like B in the system would be sufficient to cause many false alarms, because it would lead to rejecting many samples X claimed to belong to some other user A whose typing habits are steadier. In our requirement, the typing habits of B (that is, m(B)) are not used. Thus, the peculiarities of a particular user cannot affect the behavior of the whole system.

To test the performance of our approach to user authentication, we did the following experiment. The 40 people who provided 15 samples are known as legal users by the authentication system. Each sample S of each legal user U is used as a new sample, that should hopefully be recognized as belonging to U on the basis of the other 14 samples that form U's typing profile. Moreover, S is used to attack all other legal users in the system. When S, belonging to U, is used to attack another legal user U1, U's profile is temporarily removed from

Table III. Experimental Results in User Authentication for Different Distance Measures R and A[a]

Adopted Distance Measure	R_2	$R_{2,3}$	$R_{2,4}$	$R_{2,3,4}$	A_2	$A_{2,3}$	$A_{2,4}$	$A_{2,3,4}$
No. of passed impostors	563	324	279	199	590	335	366	331
No. of false alarms	50	32	41	41	92	80	84	79
IPR (%)	0.125	0.072	0.062	0.044	0.1311	0.0744	0.081	0.0736
FAR (%)	8.333	5.333	6.833	6.833	15.333	13.333	14.0	13.166

[a]Attempted attacks: 450,000; attempted legal connections: 600.

the system before running the authentication procedure, so as to make U, and his/her typing habits, completely unknown to the system. Every legal user is also attacked using the samples of the 165 individuals who provided only one sample. Hence, on the whole, the system is tested with 600 legal connection attempts and with 450,000 impostors' attacks brought by $40 + 165$ individuals ($600 \cdot 39 \cdot 15 + 165 \cdot 40 \cdot 15 = 450,000$). Readers may notice that by removing the profile of a user whose sample is used to attack another user, we are able to simulate the worst conditions for the system. In fact, each impostor's attack is now brought by someone *from the outside world*—someone whose typing habits are completely unknown to the system.[8]

As in the case of classification, here also we have to choose the distance measure to use in the experiments. We start by showing in Table III the outcomes reached by using the four R and A measures that provided the best outcomes in user classification (we have, of course, also tested the other measures, such as, e.g., $R_{3,4}$ and A_4, which, however, provide worse results, as we already have seen for the case of classification).

From Table III, we again see that R measures perform better than A measures. In particular, every A measure provides a FAR that is about twice as bad as the corresponding R measure and an IPR which is slightly higher than the one provided by the corresponding R measure. Using distances computed by summing together the values returned by R and A measures leads to the authentication results of Table IV (also in this case, for brevity, we omit the results obtained using $R_{2,4}$; they are very similar to those obtained using $R_{2,3}$). Outcomes of Table IV clearly show that the combination of R and A measures provides a better authentication accuracy. For example, when measure $R_{2,3,4}$ is summed to any of the four A measures taken into consideration (Table IVc), on average, the outcomes improve roughly twice w.r.t. the corresponding results provided by $R_{2,3,4}$ alone. We are thus able to reach an IPR of less than one false positive out of more than 3400 attempts (in the worst case), and a FAR of less than one false negative out of 25 attempts.

Measures used in the experiments can be combined together further, by using them as a set of filters applied one after the other. As a consequence, a sample X claimed to belong to user U is acknowledged as belonging to U only if it passes the authentication rule *for each* of the distance measures in the set. For example, when the adopted distance measure is, e.g., the set $\{R_2 + A_2, R_2 + A_{2,3,4}\}$, we mean that in order for a sample X to be authenticated as belonging

[8]If profiles of attacking users were not removed from the system, an attacking sample would very likely be recognized as belonging to the attacking user, and not to the user under attack.

Table IV. Experimental Results in User Authentication for Different R and A Measures Combined[a]

a				
Adopted Distance Measure	$R_2 + A_2$	$R_2 + A_{2,3}$	$R_2 + A_{2,4}$	$R_2 + A_{2,3,4}$
No. of passed impostors	360	272	260	237
No. of false alarms	36	34	37	41
IPR (%)	0.08	0.0604	0.0578	0.0527
FAR (%)	6.0	5.6667	6.1667	6.8333
b				
Adopted Distance Measure	$R_{2,3} + A_2$	$R_{2,3} + A_{2,3}$	$R_{2,3} + A_{2,4}$	$R_{2,3} + A_{2,3,4}$
No. of passed impostors	235	205	154	185
No. of false alarms	26	24	25	30
IPR (%)	0.0522	0.0456	0.0342	0.0411
FAR (%)	4.3333	4.0	4.1667	5.0
c				
Adopted Distance Measure	$R_{2,3,4} + A_2$	$R_{2,3,4} + A_{2,3}$	$R_{2,3,4} + A_{2,4}$	$R_{2,3,4} + A_{2,3,4}$
No. of passed impostors	124	78	95	131
No. of false alarms	19	23	22	23
IPR (%)	0.0276	0.0173	0.0211	0.0291
FAR (%)	3.1667	3.8333	3.6667	3.8333

[a]Attempted attacks: 450,000; attempted legal connections: 600.

Table V. Experimental Results in User Authentication for Different Filters Using R and A Measures[a]

Set of distance measures Applied as consecutive filters	$\{R_2 + A_2$, $R_2 + A_{2,3}$, $R_2 + A_{2,4}$, $R_2 + A_{2,3,4}\}$	$\{R_{2,3} + A_2$, $R_{2,3} + A_{2,3}$, $R_{2,3} + A_{2,4}$, $R_{2,3} + A_{2,3,4}\}$	$\{R_{2,3,4} + A_2$, $R_{2,3,4} + A_{2,3}$, $R_{2,3,4} + A_{2,4}$, $R_{2,3,4} + A_{2,3,4}\}$
No. of passed impostors	83	74	22
No. of false alarms	55	38	29
IPR (%)	0.0184	0.01640	0.00489
FAR (%)	9.1667	6.3333	4.8333

[a]Attempted attacks: 450,000; attempted legal connections: 600.

to user U, X must have been authenticated as belonging to U using the distance measure $R_2 + A_2$, *and* using the distance measure $R_2 + A_{2,3,4}$. Of course, we may expect the system IPR to improve and the corresponding FAR to worsen, since every sample under analysis has to pass more authentications steps. Table V shows the outcomes of this last experiment for three different sets, constructed with the measures used in Table IV. When $R_{2,3,4}$ is used together with A_2, $A_{2,3}$, $A_{2,4}$, and $A_{2,3,4}$ (last column of the table) accuracy dramatically improves. FAR still remains below 5% (that is, less than an authentication error out of 20 legal attempts), whereas the IPR shrinks of about five times, on average, reducing to less than one undetected intrusion out of 20,000 attempted attacks.

As a final remark, we observe that we also tried to take into consideration five-graphs, but without any further improvements in the outcomes. Hence, our experiments have been limited to digraphs, trigraphs, and four-graphs. It is very likely that longer *n*-graphs are not effective because a relatively limited number of them are shared, on average, between any two samples available. However, we believe that longer samples would make five-graphs useful to further improve the accuracy of our approach.

Table VI. Experimental Results in User Identification[a]

Adopted Distance Measure	k = 0.66		k = 0.75		Equal Error Rate (EER) (%)
	% of Errors on Unknown	% of Errors on Users	% of Errors on Unknown	% of Errors on Users	
$R_{2,3,4} + A_2$	0.1924	1.333	0.3762	0.8333	≈0.5 (k = 0.8)
$R_{2,3,4} + A_{2,\{3,4\}}$	0.1147	2.5	0.2527	1.6666	≈0.95 (k = 0.93)

[a] 450,000 cases of samples coming from unknown individuals; 600 cases of samples coming from known users.

5.3 User Identification and the Crossover Accuracy

Readers may have noticed that the authentication method described in the previous section is, in fact, an identification task, since a new sample X declared to belong to user A is compared not only to A's samples, but to the samples of all users known to the system. When the system authenticates X, it is merely stating that user A has been identified as the one who provided X. When the system rejects X, it is just answering "unknown." Hence, we may easily reformulate and generalize the rule used in the previous section in terms of an identification task: an incoming sample X is attributed to user A if, and only if, for any other user B and for some positive constant $k \leq 1$, the following holds:

(1) $md(A,X) < md(B,X)$;
(2) $md(A,X) < m(A) + k \cdot [md(B,X) - m(A)]$

If a user A, meeting the above rules, does not exist, X is labeled as "unknown." In the previous section, we used $k = 0.5$, since, within authentication, IPR must be kept small. However, if identification is not done for security purposes, one may want to reach a better trade-off between system IPR and FAR. In such cases, it is possible to use greater values for k, such as, e.g., $k = 0.75$ or $k = 0.66$.[9] As a consequence of the above observation, in the rest of the paper the terms *authentication* and *identification* will be used interchangeably, when referring to our system (we will often prefer to speak of *authentication*—with the corresponding IPR and FAR—as we have mainly in mind security applications).

Table VI illustrates the outcomes in user identification for two different values of k and for two of the distance measures used in the previous section: the one that provides the best FAR, $R_{2,3,4} + A_2$, and the one that reaches the best IPR, the set $\{R_{2,3,4} + A_2, R_{2,3,4} + A_{2,3}, R_{2,3,4} + A_{2,4}, R_{2,3,4} + A_{2,3,4}\}$ (and that, for the sake of conciseness, we will indicate as $R_{2,3,4} + A_{2,\{3,4\}}$ in the rest of the paper). The "% of errors on unknown" is the percentage of cases in which a sample from an "unknown" is falsely attributed to one of the users of the system. The "% of errors on users" is the percentage of cases in which a samples belonging to some user is not identified as belonging to that user. These two are simply the IPR and FAR of authentication.

The "Equal Error Rate" (EER), (last column of Table VI), is also known in the literature as the "crossover accuracy," and refers to the case when the IPR

[9] Note that for $k = 1$ we get the classification rule introduced in Section 5.1.

Table VII. Experimental Results in User Authentication for $k = 0.45$[a]

Adopted Distance Measure ($k = 0.45$)	$d = R_{2,3,4} + A_2$	$d = R_{2,3,4} + A_{2,\{3,4\}}$
No. of passed impostors	66	4
No. of false alarms	28	40
IPR (%)	0.0147	0.000889
FAR (%)	4.6666	6.6666

[a]Attempted attacks 450,000; attempted legal connections: 600.

and FAR of the system are (at least roughly) equivalent. EER can be seen as a rough estimate of the relative performance of different biometric techniques and systems, although, in general, systems are not necessarily tuned to set FAR = IPR = EER, since different applications may require a different trade-off between FAR and IPR.

Physiological biometrics have in general better-crossover accuracies than behavioral ones. For example [Mansfield et al. 2001, Ruggles 2002], iris scan reaches a 1:131,000 crossover accuracy, which shrinks to 1:500 for fingerprint analysis and hand geometry. Behavioral biometrics such as voice and signature dynamics have a crossover accuracy of about 1:50. According to the outcomes reported in the EER column of Table VI, our approach to keystroke analysis of free text shows a crossover accuracy varying from slightly more than 1:100 to 1:200.

The choice of an appropriate value for k is a very simple and effective way of reaching the required trade-off between then percentage of false negatives and false positives shown by the system, on average. As we have seen in this section, values of k sufficiently greater than 0.5 are likely not to be used for security applications. Of course, values for k smaller than 0.5 can be also adopted, in this way enforcing the security of the system at the cost of generating more false alarms. Table VII shows the outcomes of the experiments in user authentication for the distance measures used in this section with $k = 0.45$. The outcomes reached by $R_{2,3,4} + A_{2,\{3,4\}}$ are particularly interesting: by accepting a FAR of 6.66% (one error every 15 legal connections, on average), we reach an IPR of 0.000889%: less than one unspotted impostor out of 110,000 attacks.

6. MORE EXPERIMENTS AND DISCUSSION

With the aid of more experiments, in this section we will study the behavior of our method with respect to the number, length, and composition of the available samples. We will also say more on the properties of R and A measures and will discuss the scalability of our approach.

Recall that all the results of the previous section have been obtained using samples of an average length varying from 700 to 900 characters, with fourteen such samples in each user's profile. However, one may want to know how the overall performance of our method relates to the size and number of typing samples available. We will discuss these issues in this section, using the two distance measures we already adopted in the experiments on user identification: $R_{2,3,4} + A_2$ (best FAR for $k = 0.5$), and the set $R_{2,3,4} + A_{2,\{3,4\}}$ (best IPR for $k = 0.5$). All the experiments of this section will be performed with $k = 0.5$. We anticipate that using any of the other measures seen before will lead to

Table VIII. Experimental Results in User Authentication for Different Sizes of Users' Profiles

No. of Samples in Users' Profiles	$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$	
	IPR (%)	FAR (%)	IPR (%)	FAR (%)
2	0.2622	20.0788	0.1276	26.6648
4	0.09479	10.0119	0.04579	13.8654
6	0.05822	6.8703	0.02609	9.7594
8	0.04303	5.2985	0.01733	7.7751
10	0.03477	4.4097	0.01209	6.5707
12	0.029	3.8498	0.00809	5.7308
14	0.0276	3.1667	0.00489	4.8333

Table IX. Accuracy Improvement in User Authentication w.r.t. the Size of Users' Profiles

Δ of Samples in Users' Profiles	$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$	
	Δ IPR (%)	Δ FAR (%)	Δ IPR (%)	Δ FAR (%)
$2 \rightarrow 4$	-64.8	-50.1	-64.1	-48.0
$3 \rightarrow 6$	-58.2	-48.3	-62.4	-46.1
$4 \rightarrow 8$	-54.6	-47.1	-62.2	-43.9
$5 \rightarrow 10$	-51.5	-45.7	-64.0	-42.4
$6 \rightarrow 12$	-50.2	-44.0	-69.0	-41.3
$7 \rightarrow 14$	-44.0	-46.9	-76.7	-43.8

very similar conclusions. The experiments presented in the section will provide meaningful insights on the behavior of our method in real applications.

6.1 Number of Typing Samples in Users' Profiles

Table VIII shows the system outcomes in user authentication when only a subset of the samples in each user's profile is used[10] (note that the last line of the table illustrates the same outcomes shown in Tables IVc and V for the corresponding measures). Quite obviously, system accuracy increases together with the number of involved samples. When a new sample X is compared against the samples in the profile of some user U, n-graphs in X are compared with corresponding n-graphs typed by U in different moments, situations, and contexts. Hence, the larger the number of user's samples, the more meaningful is likely to be md(U, X).

By looking at the outcomes of Table VIII, it is possible to notice that, roughly, the system FAR and and IPR appear to halve when the number of samples in users' profiles doubles. We highlight this situation in detail in Table IX. Each row of the table shows the percentage of improvement of FAR and IPR for the corresponding measure and number of samples involved in the experiment. For $d = R_{2,3,4} + A_2$, IPR and FAR improvement is about 50%, on average, at each profiles' size doubling step. However, we also note that the amount of

[10]In the table, we only report the percentages because the total number of attempted attacks and legal connections change with each experiment. For example, when each user's profile contains two samples, we can simulate 3,150,000 attacks and 54,600 legal connections, by extrapolating in all possible ways two samples out of fifteen. With eight samples in users' profiles we can simulate 193,050,000 attacks and 1,801,800 legal connections.

improvement decreases slightly as more and more users' samples are added to the profiles. The reason is that new samples in users' profiles only partially contribute with new information on the typing habits of users w.r.t. the information already present and such information becomes more and more marginal as users' profiles enlarge. The outcomes for $R_{2,3,4} + A_{2,\{3,4\}}$ are different, since IPR improvement increases as more users' samples are available. This is due to the combined effect of the various distance measures used together as cumulative filters. Each distance is able to grasp slightly different aspects of the typing habits of legal users, so as to rule out impostors' samples, and such ability improves as more information is available. Of course, the presence of multiple filters may cost some additional errors on legal connections. The comparison of the IPR and FAR columns of Table IX provides experimental evidence of the effectiveness of using multiple filters like $R_{2,3,4} + A_{2,\{3,4\}}$. Corresponding numbers in the two FAR columns are similar and move in the same direction, whereas in the IPR columns they diverge quickly as more users' samples are available. Hence, we may conclude that the use of cumulative filters provides more gain in the system IPR of what is lost in the corresponding FAR. This becomes more and more evident as the size of users' profiles increases.

Numbers in Table IX suggest that there is still room left for improvement in our experimental setting. For example, we may reasonably expect improvement of the IPR of about 40% for $d = R_{2,3,4} + A_2$ when moving from 14 to 28 samples in users' profile. Nonetheless, there will come a point when adding more samples to users' profiles will no longer provide any significant improvement of accuracy: a sort of "saturation" of the information contained in the profiles that will possibly be overcome only by using longer samples, as we discuss below.

6.2 Length of the Typing Samples

System accuracy is, of course, related to the size of the typing samples. In general, the longer two samples under comparison, the higher the chances that they share a large number of n-graphs and, hence, the more meaningful the computed distance between them. This is particularly true in the case of samples of free text: if not sufficiently long, they may not even share any n-graph.[11]

Table X shows the outcomes of the authentication experiments using all the available samples in each user's profile, but using samples whose length is $\frac{1}{4}$, $\frac{1}{2}$, and $\frac{3}{4}$ of the original samples, both for the samples in users' profiles and for the sample to be authenticated (the first part of each sample has been used in the experiments). Again, the last line of the table refers to the experiments performed with samples of full length, as reported in Section 5.2 on authentication.

It is clear that longer samples, in general, share more different n-graphs and, hence, provide better accuracy than shorter samples. Even more than in the previous section, system IPR and FAR greatly improve as samples length doubles. We show this in Table XI, which is equivalent to Table IX w.r.t. the

[11]One may observe that even the number of occurrences of a given n-graph is meaningful. However, in Bergadano et al. [2002] we showed, at least for the case of R_3, that having many different n-graphs is by far more important than having them repeated many times.

Table X. Experimental Results in User Authentication for Different Length of the Samples

Length of Samples	$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$	
	IPR (%)	FAR (%)	IPR (%)	FAR (%)
1/4	0.3951	29.1667	0.1762	36.0
2/4	0.1011	11.8333	0.0633	15.5
3/4	0.0504	6.6667	0.0289	9.3333
4/4	0.0276	3.1667	0.00489	4.8333

Table XI. Accuracy Improvement w.r.t. Samples Length

Δ of the Length of Samples	$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$	
	Δ IPR (%)	Δ FAR (%)	Δ IPR (%)	Δ FAR (%)
$1/4 \rightarrow 2/4$	-64.8	-50.1	-64.1	-48.0
$2/4 \rightarrow 4/4$	-72.7	-73.2	-92.3	-59.1
$4/4 \rightarrow 8/4$	-91.2	-84.4	-90.8	-85.3

variation of samples length. (We will explain the last row of the table later). Even in the case of samples length, we may expect, at least in principle, a saturation of the information contained in the samples as their length increases, so that beyond a certain size, no greater accuracy can be reached using longer samples. However, in the case of our experiments, such point of saturation seems very far from being reached. In fact, from the first two rows of Table XI, we see that the percentage of improvement is even larger in the second than in the first row.

One may well wonder what happens for even longer samples. A partial answer is provided by the following experiment. Samples of each user have been merged together in pairs, so that from the first fourteen samples from each user, we may form seven long samples, each one having an average length of about twice an original sample (in this experiment, the last sample of each user cannot be paired with another sample and remains unused). Now we have seven long samples for each user and we may repeat the experiments in user authentication to see how the increased length of the samples affects the outcomes. New results are reported in Table XII. To help with the comparison, the first two columns of the table illustrate the outcomes when using the original, plain samples,[12] so that we may notice the improvement achieved by using samples of double length. In particular, in the last row of the first part of Table XII, we see that the use of six samples of double length improves the IPR of about ten times and the FAR of about 6 times (the exact percentage of improvement is reported in the last row of Table XI). However, note that the whole amount of information involved in the experiment with seven samples of double length

[12]It is possible to notice that the IPR outcomes of the experiments with plain samples are not exactly equivalent to those reported in Table VIII. This is due to the fact that, when performing the experiments with the long samples, impostors samples cannot be used, as their length is about only a half of that of long samples. Hence, only long samples from other users can be used to simulate impostors' attacks. For this reason we had to repeat the experiments with plain samples without using impostors' samples. Readers may however note that IPR outcomes of Tables VIII and XII (for plain samples) are quite similar. Of course, FAR outcomes are not affected by the presence or absence of impostors' samples.

Table XII. Experimental Results for Different Sizes of Users' Profiles and Length of Samples[a]

Samples in Users' Profiles	Plain Samples				Double-Length Samples			
	$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$		$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$	
	IPR	FAR	IPR	FAR	IPR	FAR	IPR	FAR
2	0.2662	20.0788	0.1296	26.6648	0.0702	6.1667	0.0379	9.119
3	0.1403	13.2862	0.0710	18.0998	0.0306	3.1607	0.0152	4.75
4	0.0958	10.012	0.0479	13.8654	0.0186	2.2143	0.00916	3.0476
6	0.0592	6.8703	0.0286	9.7594	0.00523	1.0714	0.00262	1.4286

Samples in Users' Profiles	Triple-Length Samples			
	$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$	
	IPR	FAR	IPR	FAR
2	0.0372	3.6667	0.0218	4.5833
3	0.00641	1.25	0.00256	1.75
4	0.0	0.5	0.0	1.0

[a]As usual, values of IPR and FAR columns are percentages.

is about the same as involved when using all fifteen original samples, only arranged in a different way. Hence, it is correct to compare the outcomes reached using all available samples of double length with the outcomes reached using all original samples (see, e.g., the last row of Table X). The improvement due to double length samples is still evident. The IPR improves between two and five times for the two measures used in the experiments and the FAR improves about three times for both measures.

By merging three consecutive samples, we obtain even better results, reported in the second part of Table XII. When all available information is used (in order to form five samples of triple length: four of them to form users' profiles, and the remaining one that must be authenticated), we reach an impressive IPR = 0% for both measures used in the experiments, with corresponding FARs better than in any other experiment described so far.

Although good, it is, however, only fair to note that the outcomes reached using double-, or triple-length samples are less meaningful than those obtained using the original samples. In fact, with six double-length samples in users' profiles, it is possible to test the system with only $40 \cdot 7 = 280$ legal connections and $280 \cdot 39 \cdot 7 = 76{,}440$ intrusions. Using triple-length samples, the system can be tested with only $40 \cdot 5 = 200$ legal connections and $200 \cdot 39 \cdot 5 = 39{,}000$ intrusions. In both cases, each user is attacked by only 39 different individuals. Nonetheless, the experiments of this section show clearly that the accuracy of our method is strongly related to the length of the available samples. Moreover, given a certain amount of information about the typing habits of users, it seems clearly better to organize it as a small number of longer samples, instead of using many shorter samples (see also the next subsection).

One may, of course, wonder why not just merge all available samples of a user U, in order to form a unique very long sample that represents the entire U's profile. The reason for not doing so is that at least two samples in U's profile are necessary to compute the mean distance of the samples in the profile (that is, m(U)) and to perform the authentication step, as described in Section 5.2. Moreover, more samples in U's profile make m(U) more meaningful, and the authentication/identification step more accurate.

Table XIII. Experimental Results for Test Samples of Different Length

Length of Test Samples	$d = R_{2,3,4} + A_2$		$d = R_{2,3,4} + A_{2,\{3,4\}}$	
	IPR (%)	FAR (%)	IPR (%)	FAR (%)
1/4	0.04	28.16	0.0148	31.83
2/4	0.0275	13.0	0.0086	16.66
3/4	0.028	7.0	0.0079	10.5
4/4	0.0276	3.16	0.00489	4.8333

We conclude this section by showing what happens to system accuracy when only the length of the typing sample under analysis varies, while the size of users' profiles remains fixed. Table XIII shows the outcomes of this experiment with fourteen original samples in users' profiles, and with only the first, second, and third quarter of the test sample used (the last line of the table illustrates the outcomes found in Section 5.2 for the corresponding measures). From Table XIII (it is useful to compare it with Table X) it appears that the length of the sample under analysis is especially important to have a low FAR, while the IPR remains pretty small even when only the first quarter of the sample under analysis is used by the system. In practice, this means that if X comes from an impostor pretending to be U, md(U,X) tends to remain constant, or to increase, as the length of X increases. If X is being entered by U, md(U,X) tends to shrink as more and more keystrokes are entered by the user. In practical applications, the incoming sample X can be analyzed at regular intervals and md(U,X) can be recomputed. By observing md(U,X) to increase or decrease, one may gather meaningful information about the identity of the individual typing at the keyboard.

6.3 On the Usefulness of Short Samples

Roughly speaking, a typing sample can be considered to be a sequence of keystrokes that produces a coherent piece of text and that, more or less, is entered continuously. For example, a secretary copying a long document is producing a typing sample, as well as an individual writing an email. On the other hand, an URL typed into the browser address toolbar is a (very short) typing sample too, as well as a Unix command, for example, "ls -lR mydir > myfileslist."

Hence, in real situations, typing samples stemming from individuals using computers may be of largely different lengths. Of course, a continuous monitoring of the keyboard may gather all such samples. However, what should be done with them, especially with the shortest ones? In Section 6.2, we saw that there is a direct relation between the length of the typing samples and system accuracy, so that very short samples are likely to be useless to authenticate/identify someone, as well as worthless to be used as typing samples in users' profiles. On the other hand, long samples are likely to be infrequent.

An obvious solution to handling short samples is to merge them, in order to form samples long enough to be useful. It remains to be proved that the level of accuracy that can be attained with such samples is comparable to the use of samples of similar length each produced in a unique typing session, as in the experiments described so far.

Table XIV. Experimental Results in User Authentication Using "Scrambled" Profiles[a]

Adopted Distance Measure	$d = R_{2,3,4} + A_2$	$d = R_{2,3,4} + A_{2,\{3,4\}}$
No. of passed impostors	60	33
No. of false alarms	29	48
IPR (%)	0.0133	0.0073
FAR (%)	4.8333	8.0

[a]Attempted attacks: 450,000; attempted legal connections: 600.

To test the usefulness of the above solution, we performed the following experiment. Each of the 14 samples in each user's profile was split into pieces of 25 characters entered consecutively. These *short* samples were then grouped randomly in order to form 14 samples (let's call them "scrambled" samples) of a length similar to the original ones. As a consequence, each scrambled sample was composed of a set of short samples typed in different days, possibly in different months. Each short sample was used only once to form a scrambled sample and then removed.[13] For each user, the sample not involved in the scrambling process was left unchanged and was used as test sample. Impostors samples are also left unchanged. The whole process is repeated for each of the fifteen samples provided by the legal users.

Table XIV illustrates the results for this experiment. Outcomes are only slightly worse than those of Section 5.2 and we believe that this provides a good evidence of the fact that, in real applications, short samples may be merged to make them more useful. It must be observed that the experiment we have performed produces a situation that is very unfavorable (and even very artificial) for the system. If fact, it must use scrambled samples made of pieces that may have been produced at a distance of weeks or even months, possibly on different keyboards and systems, and two adjacent short samples may be such that the second one has been entered by the user *before* the first one. In real situations, short samples gathered together to form a long sample will be often produced by the users shortly after one another, such as in the case of a user that first enters a few Unix commands, then writes an email, and then an Internet address in the navigation toolbar of the browser, and so on.

6.4 Experimental Properties of R and A Measures

Consider the 600 samples collected from 40 legal users in our experiments. Table XV illustrates, for different distance measures, the mean distances and corresponding standard deviations between any two samples of the same user and between any two samples from different users. These values show why our system recognizes users on the basis of their typing rhythms: on average, two samples from the same user have a smaller distance than two samples

[13]It must be noted that splitting and rejoining samples in this way produces a loss of information. When merging two short samples, the digraph made of the last character of the first short sample and the first character of the second short sample has a meaningless duration. Since short samples are long 25 characters, the loss of information due to such digraphs is about 1/25 of the total number of available digraphs.

Table XV. Means and Standard Deviations for Distances between Samples from the Same (4200 Comparisons) and Different Users (175,500 Comparisons)

Distance Measure	Same user	Different users
	Mean / st. dev.	Mean / st. dev.
R_2	0.390065 / 0.052713	0.521580 / 0.060695
R_3	0.483910 / 0.047813	0.580740 / 0.046899
R_4	0.509219 / 0.057563	0.600801 / 0.054895
A_2	0.460609 / 0.069778	0.629887 / 0.077474
A_3	0.519802 / 0.070180	0.665812 / 0.084802
A_4	0.505551 / 0.084802	0.662948 / 0.101075
$R_{2,3,4} + A_2$	1.843803 / 0.176803	2.333008 / 0.179031

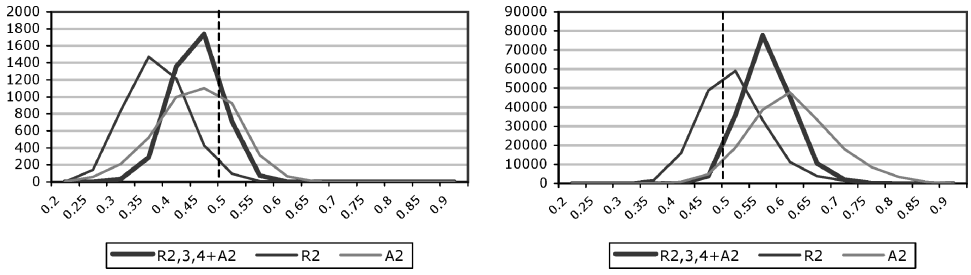

Fig. 2. Distributions of distances between samples from the same user (left) and from different users (right) in the interval [0–1].

from different users. It is interesting to compare the values of Table XV with the outcomes in user classification obtained using the corresponding distances (Table I). For example, R_2 works better than R_3 or R_4 and, in fact, the difference between the mean distance of samples from the same user and the mean distance of samples from different users is larger in the case of R_2 than in the case of R_3 or R_4. The same occurs for A measures. Note, in particular, the mean values for A_2. Their difference is even a little bit larger than in the case of R_2, but the classification results using A_2 are worse than those reached using R_2. However, observe that the standard deviations for A_2 are greater than those for R_2. This higher variability accounts for a larger number of classification errors when using A_2. Standard deviations are even larger for A_3 and A_4 and, in fact, these distances show a lower discriminating power than R_2, R_3,R_4, and A_2.

The discriminating power of R_2, A_2, and $R_{2,3,4} + A_2$ is also depicted in Figure 2. The left part of the figure shows the distributions in the interval [0–1] of all distances between any two samples from the same user computed with the three measures. On the right are the corresponding distributions for all distances between any two samples from different legal users. Note that distances computed using $R_{2,3,4} + A_2$ (which is the sum of four distances) have been divided by four in order to allow the comparison with R_2 and A_2. The two graphs have different scales, since there are $15 \cdot 14 \cdot 40/2 = 4200$ comparisons between samples from the same user and $600 \cdot 599/2 – 4200 = 175{,}500$ comparisons between samples from different legal users. The shape and position of the distributions in the two graphs show the clear separation between samples

from the same user and samples from different users operated by the three distances. Note also that distances computed using $R_{2,3,4} + A_2$ concentrate in a narrower distribution than in the case of R_2 and A_2, which accounts for the better performance of $R_{2,3,4} + A_2$ w.r.t. R_2 or A_2 used alone.

The partial overlapping of corresponding distributions in the two graphs is mostly only apparent, since they are traced over different scales. For example, in the case of $R_{2,3,4} + A_2$, in the interval [0.5–0.55] (actually, in the interval [2.0–2.2]) fall 711 distances between samples from the same user and 35,817 distances between samples from different users. Moreover, even when, for users A and B, it is the case that d(A1,A2) $\simeq$ d(A1,B1), with A1, A2 coming from A, and B1 from B, this does not imply that A and B have similar profiles. This only happens if d(A1,A2) $\simeq$ d(A1,B1) for most of the typing samples of A and B.

We conclude with an observation on A measures. We saw in Section 3.2 that they allow to compare all *n*-graphs shared between two samples, even when some of the *n*-graphs occur so infrequently that a meaningful mean and standard deviation cannot be computed (so that most of the techniques described in Section 2 could not be easily applied). Nonetheless, one may wonder how A measures perform w.r.t. other measures that rely on the absolute typing speed of *n*-graphs. An obvious term of comparison is the Euclidean distance; therefore, we repeated the experiment in user classification described in Section 5.1 in the case of digraphs, but using the Euclidean distance. Following the notation we used for R and A measures, we computed E_2(**S1**, **S2**) for any two samples **S1** and **S2** from legal users (actually, E_2(**S1**, **S2**) is the Euclidean distance between digraphs shared by **S1** and **S2**, divided by the number of shared digraphs, so that we can compare distances between pairs of samples sharing a different number of digraphs). E_2 provides a classification error of 68,83%: 413 samples are not correctly classified, over a total of 600 attempted classifications. We recall that using A_2 we got a classification error of 7.33%. In Monrose and Rubin [1997], authors reach a classification accuracy of 21.3% (hence, an error rate of 78.7%) comparing typing samples of free text with the Euclidean distance (though, of course, they use a different classification rule, as seen). It is interesting to note that the Euclidean distance on our data set performs more or less in the same way.

6.5 On the Scalability of Our Experiments

The scalability of our approach to keystroke analysis is, first of all, related to the computational costs required to authenticate a new sample. Computing the distance between two samples is expensive, as it involves ordering the samples w.r.t. the typing speed of the graphs they contain. In our experiments, comparing a new sample against 40 profiles containing 14 samples each one, and computing distances for digraphs, trigraphs, and four-graphs, requires about 140 sec on a Pentium IV at 2.5 GHz. Admittedly, our programs are far from being optimized. It is very good that the computation of distances can be strongly parallelized. For example, with three processors, we may compute in parallel distances w.r.t. digraphs, trigraphs, and four-graphs, sensibly shrinking the time needed to handle new samples.

In our system, scalability may also be limited by the fact that, if two users have very similar typing habits, both could raise a false alarm when they are accessing their own account. The larger the number of legal users, the higher the chances of such a situation. Actually, when very many users are involved, the problems of both computational costs and very similar profiles can be limited if our method is used for true authentication purposes (that is, when the sample under analysis comes with a claimed identity). To limit computational costs, it is not necessary to compare a new sample with all users' profiles, but just with a subset of manageable size.[14] Subsets containing a number of profiles and samples close to the one used in our experiments will very likely provide a similar level of accuracy. When two legal users (say, A and B) of a system have very similar typing habits, false alarms can be avoided by excluding the profile of B when performing the authentication of a sample claimed to belong to A. Clearly, if B attacks A, he will be treated as an unknown individual and will have some more chances of fooling the system. However, in order for B to judge worth attacking A, B *must know* that his typing habits are similar to those of A. Without this knowledge (which may not be easy to acquire), B is left to randomly choose the user to attack. We note that the proposed solution will not work well if many, or most of the users in the system, have similar profiles, and one may wonder how the space of profiles fills in as the number of users increases.

Only extensive experiments could provide an answer to the above question, but we can make some remark on the basis of the samples of the 40 legal users of our experiments. Let mdu(A,B) be the "distance between users A and B," that is, the mean distance between any two samples of A and B. For instance, if the samples of A and B are, respectively, A1, A2, and B1, B2, we have mdu(A,B) = [(d(A1,B1) + d(A1,B2) + d(A2,B1) + d(A2,B2)]/4. mdu(A,B) gives an idea of the distance we may expect between any sample of A and B's profile, or vice versa. In fact:

$$[md(A,B1) + md(A,B2)]/2 =$$
$$\{[d(A1,B1) + d(A2,B1)]/2 + [d(A1,B2) + d(A2,B2)]/2\}/2 =$$
$$[d(A1,B1) + d(A1,B2) + d(A2,B1) + d(A2,B2)]/4 = mdu(A,B)$$

In general, the smaller mdu(A,B), the more similar are the typing habits of A and B, and the higher the chances that users A and B cause false alarms. However, the important information lies in the relation between m(A) and mdu(A,B). As observed in Section 5.2, m(A) gives us an idea of the distance we may expect between two typing samples provided by user A. When $m(A) \simeq mdu(A,B)$ then samples from A and B cannot be distinguished any longer. We have computed mdu(A,B) for all pairs of legal users in our experiments, for a total of $40 \cdot 39/2 = 780$ different pairs. It never happens that $m(A) \simeq mdu(A,B)$ for any two users A and B and, on average, mdu(A,B) is about 16% greater than m(A) and m(B). On the contrary, for any sample A1 belonging to A, md(A,A1) is only 5% greater than m(A), on average.[15]

[14]Of course, the subset will also contain the profile of the user the sample is claimed to belong to.
[15]Of course, in this case md(A,A1) and m(A) are computed removing A1 from A's profile.

Scalability is also related to the typing skills of users. In our experiments, if users A and B are fast typists (that is, the mean duration of n-graphs in their samples is small), mdu(A,B) is normally large (of course, mdu(A,B) is even larger if A is a fast typist and B is a slow one). On the contrary, slow typists have, in general, a small mdu between each other. We got similar results even in the experiments with fixed test [Bergadano et al. 2002]. Hence, it is likely that more false alarms must be expected if there are many unskilled typists among the legal users of the system.

7. APPLICATIONS

Being able to verify personal identity of users through the keystroke dynamics of free text means, first of all, that the profiling phase becomes much easier. In fact, the formation of users' models by means of fixed, long texts, is clearly perceived as an annoying burden and may lead people to reject the corresponding technology. Nonetheless, even in the case of free text, the registration process will have to take place in a protected environment, in order to avoid impostors being able to manipulate users' profiles through the introduction of false samples. During the profiling phase, users' identity must clearly be checked in some alternative way, and the production of typing samples should be limited to a few days, not weeks or months. Of course, users themselves could be willing to provide (part of) their typing profiles in dedicated sessions that would not take a long time (an average typist enters a text long like the samples used in our experiments in less than 5 min).

Interested readers may experiment with keystroke analysis through a simple on-line prototype of our system that can be found at www.di.unito.it/~gunetti/prototype.html. The HTML page is very similar to the one we wrote to collect volunteers' samples and uses the same Javascript to gather the timing information. People not registered on the system may, of course, only test it as impostors, pretending to be one of the legal users (for example, entering *gunetti* or *picardi* as login name). The prototype is, however, also able to build from scratch the profile of a new user, asking him a number of samples until a sufficient amount of typing information has been provided. From that moment, the system will be able to identify the user through his keystroke dynamics.[16] This will also help us to further test our method and to extend the size of our data set.[17]

Since through the typing rhythms of people we may ascertain personal identity, the natural application of keystroke analysis is in the field of security. It is obviously unrealistic to propose keystroke analysis as a plain substitute of well-established technologies, such as, for example, password-based systems. Nonetheless, authentication methods based on keystroke analysis can still be useful, as we argued in Bergadano et al. [2002]. This is even more true if free text can be used.

[16]To be registered as a new user, send an email to one of the authors.

[17]We also make available the 765 typing samples of free text that we gathered and used in our experiments. This will allow for a possible independent evaluation of our outcomes. The data set could also be useful to test other approaches to keystroke analysis. Contact the authors for more information.

7.1 Lost and Forgotten Passwords

The most obvious drawback of passwords is that they can be forgotten. This problem is especially frequent over the Web, where many services are accessible through passwords. The most common of such services is a free email account, that is often provided together with access to dedicated discussion groups and chat lines. Especially if the connection to such sites is infrequent, users tend to forget the passwords they chose.

Lost and forgotten passwords are turning into such a stringent problem, that even the press is aware of it [BBC 2000]. The New York Times on the Web site [Lee 1999] estimates that more than 1000 people forget their password to the site each week and 10–15% of its registrants are duplicates. Two much cited sources on this issue can be found on the Web. According to the Gartner Group, an organization with 2500 desktop computers can spend more than $850,000 per year resetting passwords [Novell ViewPoints]. Forrester Research, Inc. has found that password problems account for 40–80% of all IT help-desk calls.[18]

In some cases, an automatic protocol is available to help users to retrieve a lost password, but such recovery methods are well known not being very reliable. Hence, an alternative authentication method based on keystroke analysis can be provided by the site to allow a registered user to regain access to his/her account in a few minutes. The ability to deal with free text will allow the use of emails and messages composed by the users to build their typing profiles, silently and transparently (although possibly, when forming a user's profile, with the limitations we mentioned at the beginning of this section). In order to be authenticated, users will be free to enter whatever text they like, provided it is sufficiently long to allow a sufficient level of accuracy. A similar solution can also be useful to retrieve passwords of job accounts (that, fortunately, are seldom lost), when a system administrator is not immediately available. Long texts required for the authentication process (e.g., at least 1000 keystrokes, that may need a few minutes to be entered) will assure a sufficient level of accuracy and will discourage potential intruders.

When a very high level of security is needed, keystroke analysis can also be used *in conjunction* with passwords. To be authenticated, an individual must know the password *and* must provide a typing sample to be checked against the reference profile. Alternatively, the secret pass phrase itself could be used as a typing sample, provided it is sufficiently long. In such case the pass phrase would have to be entered only in environments protected from any kind of spying, since it would be too awkward to type long texts without having the entered characters displayed, as in the case of common passwords.

The above applications, of course, are meant for login time, but typing rhythms do exist throughout an entire working session, *after* an authentication phase has been passed—legally or not. Thus, such typing rhythms may still be exploited, as we discuss in the next section.

[18]It is fair to note that some people feel such statistics being overestimated. See, e.g., on the bulletin list of Internet Service Provider journal at www.isp-planet.com/marketing/2002/password_price_bol.html

7.2 Intrusion Detection

In Bergadano et al. [2002] we mentioned *Identity Confirmation* as a possible application of keystroke analysis: if an intrusion detection system notices some strange behavior of an individual using the computer,[19] it may ask the individual to provide a confirmation of his/her identity: a typing sample to be checked against the typing profile of the legal owner of the account. However, such solution is only partially satisfactory: if the suspicious individual is in fact an intruder, he/she will be aware of having being spotted and will immediately disconnect.

However, being able to deal with free text makes it possible to transparently analyze typing rhythms of individuals. This makes keystroke analysis perfectly suitable (at least in principle) *to be part* of an intrusion detection system (IDS for short). Stated boldly, if an individual using an account shows typing dynamics different from those in the typing profile of the account's owner, that individual is an impostor using the account fraudulently.[20] Essentially, this may be seen as a form of anomaly detection: the individual under observation is not behaving as expected.

Of course, things in real applications are not so easy, and keystroke analysis is useless (as probably most other techniques) if the impostor just enters something like "rm -r $*$" and logouts. On the other hand, if the intruder wants to steal information or to illegally use resources, he will try to connect to the system remaining unnoticed for as long as possible. Hence, apart from extreme situations, keystroke analysis of free text can be added to the set of tools available to detect intrusions. In particular, we believe it can be very useful to limit the number of false alarms, an endemic problem of IDSs.[21] We have observed in Section 6.5 that our approach to keystroke analysis is computationally expensive. Hence, a continuous monitoring and analysis of typing rhythms of all the users logged on the system—a true keystroke analysis in real time—could be impractical. However, such analysis can be started only on demand, when an IDS is raising an alarm.

In Axelsson [1999], the author shows that an IDS requires a FAR largely lower than 0.1% to be effective. The reason is easy to explain with an example (reworked from [McHugh 2000]): suppose that an IDS has a 100% true detection rate and a 0.01% FAR. Suppose also that we have a data set containing 5,000,000 units of analysis so that the system has to make 5,000,000 decisions and that 1 out of 50,000 units contains a true intrusion. then, as a consequence of the rates above, all true intrusions will be detected (which will be about 100),

[19]Such as some exotic Unix command run from a secretary account.

[20]Clearly, we cannot take into consideration the case where the same account can be legally used by more individuals.

[21]Especially of those performing anomaly detection, as a consequence of changes in legal users' habits [McHugh 2000]. One may observe that also typing habits change, so that keystroke analysis could itself be a source of false alarms. However, typing skills normally improve up to a certain point, and then they tend to remain stable, and users' profiles can be kept updated by simply replacing old typing samples with newer ones. Other users' habits are much more subject to change, as a consequence of the use of new applications and program versions, new computers, different duties and tasks assigned.

but together with them also about 500 false alarms will be raised. If alarms must be handled manually, a system administrator will spend most of his/her time analyzing legal situations wrongly labeled as intrusions, increasing the chances that true intrusions go unnoticed.

Suppose now that we have a typing profile of all legal users of a system and that the timing information of keystrokes issued by the users connected to the system are continuously gathered (but not analyzed). Suppose, moreover, that an intrusion alarm is raised for a particular account and that a sufficient number of keystrokes have been collected for the suspicious account in the current login session. Keystroke analysis can be used on the timing data collected to confirm/cancel the alarm before it reaches the system administrator. Consider the example in the previous paragraph and that the system performing keystroke analysis is tuned to have a 0.5% FAR and 0.5% IPR, on average. If every alarm can be analyzed by the system (i.e., keystroke analysis is possible for all accounts raising an alarm), the whole FAR of the IDS will shrink to 0.00005%, whereas the IPR will only worsen to 0.5%. Using the figures above, the number of false alarms raised will reduce to about two or three, whereas possibly all 100 intrusion will still be detected.

There are, of course, forms of intrusions where keystroke analysis is useless, but it is very likely that it will be useful in most cases of stolen or cracked passwords. For example, the timing data of keystrokes issued by users can be collected during their connection to the system, while keystroke analysis on such data can be performed overnight on all accounts accessed the previous day (or, on a particular account, immediately after log-off). Possible anomalies would be simply reported to the system administrator and, at the very least, the legal user of the account could be suggested to change his/her password. In such case, even a FAR of 0.5 or 1% would not be a serious problem: false alarms will simply make users change their passwords a bit more frequently than usual. In any case, the number of false alarms raised each day would be limited to one or two, assuming, e.g., a system with about 100 users that connect to the system once per day.

We observe that in modern systems is relatively easy to sample whatever is entered at a computer keyboard. For example, in our previous work [Bergadano et al. 2002] we used an X Window application to gather the the typing samples. In MS Windows, a keyboard "hook" can be used to log all keystrokes issued from an account, together with the relative timing data. In the case of Web connections, even a simple Javascript can be sufficient (The one we used to collect our samples is long less than 40 lines and can be seen at the URL of our on-line system). Once a sufficient amount of raw data is available, the system performing keystroke analysis can be started.

When users are providing their samples to register on the system, they, of course, know they are being monitored. However, a continuous sampling of the keystrokes issued on a computer keyboard may raise strong concerns about user's privacy [Volokh 2000]. Users must, of course, be made aware that they are under observation, but they must also understand (and hopefully accept) that every security policy must imply, in some way, a limitation of their privacy. Nonetheless, at the very least monitored text will have to be blurred, only

stored in terms of n-graphs and the corresponding timing information, and made available exclusively to the authentication/identification system.

8. CONCLUSION

In this paper we have described a method to analyze the typing rhythms of free text. The technique has been tested on a set of volunteers larger than in other experiments described in the literature, performing well at discriminating among legal users and impostors. The European Standard for Access Control (EN 50133-1) requires a commercial biometric system to have a FAR less than 1% and an IPR less than 0.001% [Polemi 2000]: we are not all that far from such numbers (and we also reached them using triple-length samples, even if we are well aware that such outcomes must be taken with a grain of salt).

Our experiments were performed without any particular form of overfitting or any specific tuning of the system on the available samples. This tuning is, however, possible and should be adopted in real applications: (1) by looking for the value of t in A measures that performs better for each legal user: this will improve the ability of the system to discriminate between legal users and impostors; (2) by choosing the value of k (as described in Section 5.3) that works better for the intended application: this will allow to find the desired trade-off between FAR and IPR. Finally, the ability to deal with typing samples of free text makes it easier to form users' profiles and to keep them updated. Samples will have to be gathered in a protected environment, so as to avoid the introduction of false samples. Users will still be free to keep on doing their normal job.

Typing dynamics is the most natural kind of biometrics that stems from the use of computers, it is relatively easy to sample, and it is available throughout the entire working session. The analysis of such dynamics can be used to ascertain personal identity and, hence, it can be helpful to improve the security of computer systems, as we have proposed in this paper.

ACKNOWLEDGMENTS

Many thanks to the volunteers of our Department who contributed to our research. Thanks also to the anonymous reviewers who suggested many improvements to the paper.

REFERENCES

ASHBOURN, J. 2000a. *Biometrics: Advanced Identity Verification*. Springer.

ASHBOURN, J. 2000b. The distinction between authentication and identification. Paper available at the Avanti Biometric Reference Site. (homepage.ntlworld.com/avanti).

AXELSSON, S. 1999. The base-rate fallacy and its implications for the difficulty of intrusion detection. In *Proc. of the 6th ACM Conf. on Computer and Comm. Security*. Singapore.

BBC 2000. You must remember this... that and other. BBC News. Also available at: news.bbc.co.uk/hi/english/uk/newsid_720000/720976.stm.

BERGADANO, F., GUNETTI, D., AND PICARDI, C. 2002. User authentication through keystroke dynamics. *ACM Transactions on Information and System Security 5*, 4.

BERGADANO, F., GUNETTI, D., AND PICARDI, C. 2003. Identity verification through dynamic keystroke analysis. *Journal of Intelligent Data Analysis 7*, 5.

BLEHA, S., SLIVINSKY, C., AND HUSSEIN, B. 1990. Computer-access security systems using keystroke dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence 12*, 12, 1217–1222.

BROWN, M. AND ROGERS, S. J. 1993. User identification via keystroke characteristics of typed names using neural networks. *International Journal of Man-Machine Studies 39*, 999–1014.

CLARKE, N., FURNELL, S., LINES, B., AND REYNOLDS, P. 2003. Using keystroke analysis as a mechanism for subscriber authentication on mobile handsets. In *Proc. 18th Conf. on Information Security. (IFIP/SEC 2003)*. Kluwer, Athens, Greece.

DOWLAND, P., SINGH, H., AND FURNELL, S. 2001. A preliminary investigation of user authentication using continuous keystroke analysis. In *Proc. 8th IFIP Annual Working Conf. on Information Security Mangement and Small System Security*. Las Vegas, Nevada.

DOWLAND, P., FURNELL, S., AND PAPADAKI, M. 2002. Keystroke analysis as a method of advanced user authentication and response. In *Proc. of the 17th IFIP/SEC 2002 Conference*. Kluwer, Cairo, Egypt.

FURNELL, S., MORRISSEY, J., SANDERS, P., AND STOCKEL, C. 1996. Applications of keystroke analysis for improved login security and continuous user authentication. In *Proc. 12th Conf. on Information Security. (IFIP/SEC 1996)*. Samos, Greece.

GUNETTI, D. AND BERGADANO, F. 2002. Method and apparatus for verifying an individual's identity based on biometric keystroke properties. Italian Industrial Patent n. 1.311.278.

JOYCE, R. AND GUPTA, G. 1990. User authorization based on keystroke latencies. *Communications of the ACM 33*, 2, 168–176.

LEE, J. 1999. Forgot a password? try way2many; better on-line security has meant more passwords, and more frustrate users. The New York Times. See also at: www.azstarnet.com/public/startech/archive/081999/main.htm.

LEGGETT, J. AND WILLIAMS, G. 1988. Verifying identity via keystroke characteristics. *International Journal of Man-Machine Studies 28*, 1, 67–76.

LEGGETT, J., LONGNECKER, M., WILLIAMS, G., AND USNICK, M. 1991. Dynamic identity verification via keystroke characteristics. *International Journal of Man-Machine Studies 35*, 859–870.

MANSFIELD, T., KELLY, G., CHANDLER, D., AND KANE, J. 2001. Biometric product testing final report. Deliverable of the biometric working group, National Physical Laboratory. Available at www.cesg.gov.uk/technology/biometrics/media/Biometric%20Test%20Report%20pt1.pdf.

MCHUGH, J. 2000. Testing intrusion detection systems. *ACM Transactions on Information and System Security 3*, 4, 262–294.

MONROSE, F. AND RUBIN, A. 1997. Authentication via keystroke dynamics. In *Proc. of the 4th ACM Conf. on Computer and Communications Security*. ACM Press, New York, 48–56.

NOVELL VIEWPOINTS (GARTNER GROUP). Find the balance between network security and growing user access demands. Available at www.gartner.com/gc/webletter/novell/issue1/article1/article1.html.

OBAIDAT, M. S. AND SADOUN, B. 1997. Verification of computer users using keystroke dynamics. *IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics 27*, 2, 261–269.

POLEMI, D. 2000. Biometric techniques: Review and evaluation of biometric techniques for identification and authentication, including an appraisal of the areas where they are most applicable. Report prepared for the European Commission DG XIII—C.4 on the Information Society Technologies. Available at: www.cordis.lu/infosec/src/stud5fr.html.

RUGGLES, T. 2002. Comparison of biometric techniques. Biometric Technology, Inc. Available at www.bio-tech-inc.com/bio.htm.

UMPHRESS, D. AND WILLIAMS, G. 1985. Identity verification through keyboard characteristics. *International Journal of Man-Machine Studies 23*, 263–273.

VOLOKH, E. 2000. Personalization and privacy. *Communications of the ACM 43*, 8, 84–88.

Received November 2003; revised February 2005; accepted July 2005