Continuous Verification Using Multimodal Biometrics

Terence Sim, *Member*, *IEEE*, Sheng Zhang, *Student Member*, *IEEE*, Rajkumar Janakiraman, and Sandeep Kumar

Abstract—Conventional verification systems, such as those controlling access to a secure room, do not usually require the user to reauthenticate himself for continued access to the protected resource. This may not be sufficient for high-security environments in which the protected resource needs to be continuously monitored for unauthorized use. In such cases, continuous verification is needed. In this paper, we present the theory, architecture, implementation, and performance of a multimodal biometrics verification system that continuously verifies the presence of a logged-in user. Two modalities are currently used—face and fingerprint—but our theory can be readily extended to include more modalities. We show that continuous verification imposes additional requirements on multimodal fusion when compared to conventional verification systems. We also argue that the usual performance metrics of false accept and false reject rates are insufficient yardsticks for continuous verification and propose new metrics against which we benchmark our system.

Index Terms—Pattern recognition, biometrics, fusion, verification.

1 Introduction

For most computer systems, once the identity of the user has been verified at login, the system resources are typically made available to the user until the user exits the system. This may be appropriate for low-security environments but can lead to session "hijacking" (akin to hijacking [1]) in which an attacker targets a postauthenticated session. In high risk environments, or where the cost of unauthorized use of a computer is high, *continuous verification*, if it can be realized efficiently, is important to reduce this window of vulnerability. By this, we mean that verification is not merely used to authenticate a session at the initial login, but that it is used in a loop throughout the session to continuously authenticate the presence/participation of the user. Examples where continuous verification is desirable include aircraft cockpits, defense establishments, and in other computer processing that affects the security and safety of human lives. In such situations, the desirable default action might be to render the computer system ineffective when the authorized user is not the one controlling it.

One way to realize (an approximation of) continuous verification is to use biometrics. Biometrics verification is appealing because several of them that are easily incorporated into ordinary computer usage are *passive* (that is, they do not require the active cooperation of the user and are therefore not distracting) and they obviate the need to carry extra devices for authentication.[1] In a sense, they are always on one's "person" and perhaps a little safer than using external devices which can be separated from their carrier more easily. However, a single biometric may be inadequate for passive verification either because of noise in the observation samples or because of the unavailability of an observation at a given time. For example, face verification cannot work when frontal face detection fails because the user presents a nonfrontal pose. To overcome these limitations, researchers have proposed the use of multiple biometrics and have demonstrated increased accuracy of verification with a concomitant decrease in vulnerability to impersonation [2]. The use of multiple biometrics has led to the investigation of integrating different types of inputs (modalities) with different characteristics. Kittler et al. [3] experimented with six combination (also known as fusion, integration) methods for face and voice biometrics, using the sum, product, minimum, median, maximum, and majority vote rules. These rules are also popular for combining classifiers in general pattern classification problems and have been extenstively studied in the literature. See, for example, [4], [5].

In this paper, we describe the theory, architecture, implementation, and performance of a multimodal biometrics verification system that continuously verifies the presence of a user using a protected computer. If the verification fails, the system reacts in accordance with security policies by locking the computer and by delaying or freezing the user's processes that are executing in the operating system. Two modalities are currently used—face and fingerprint—but our theory can be readily extended to include more modalities in the future. The main thrust of our work here is that continuous verification imposes additional requirements on multimodal fusion when compared to conventional, one-time verification systems. Our secondary aim is to propose new performance metrics that are better suited for continuous verification.

- *T. Sim, S. Zhang, and R. Janakiraman are with the School of Computing, National University of Singapore, 3 Science Drive 2, 117543 Singapore. E-mail: {tsim, zhangshe, rajkumar}@comp.nus.edu.sg.*
- *S. Kumar is with General Motors India, Third Floor Creator Building, International Technology Park, Whitefield Road, Bangalore 560066, India. E-mail: sandeep.kumar1@gm.com.*

Manuscript received 1 Feb. 2006; revised 25 July 2006; accepted 29 Aug. 2006; published online 18 Jan. 2007.
Recommended for acceptance by S. Prabhakar, J. Kittler, D. Maltoni, L. O'Gorman, and T. Tan.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org and reference IEEECS Log Number TPAMISI-0084-0206.
Digital Object Identifier no. 10.1109/TPAMI.2007.1010.

1. We will use the terms "verification" and "authentication" interchangeably.

TABLE 1
Three Criteria for Continuous Verification

1) The different reliability of the different modalities must be accounted for. That is, any fusion method must factor in the reliability of each modality. For instance, fingerprint is generally considered to be more reliable than face and therefore must affect the final decision more heavily than face.

2) Older observations must be discounted to reflect the increasing uncertainty of the continued presence of the legitimate user. For example, even when an observation obtained at an earlier time had a high reliability, there is no guarantee that the legitimate user is still present at the computer console. S/he may have left without logging out, or a hijacker may have forcefully taken over. The longer the elapsed time, the more uncertain the continued presence of the user.

3) It must be possible to determine "authentication certainty" (whether the legitimate user is present or not) at any point in time, even when there are no observations in one or more modalities. In the normal course of using a continuous verification system, it is expected that some observations may be missing, e.g., when the user looks away from the camera.

By conventional, one-time verification systems we mean those systems in which authentication is performed before access to a protected resource is granted to a user. After access is given, such systems typically do not require the user to reauthenticate for continued access to the resource, at least not for a reasonably long period of time. An example of this is a secure room whose entrance is protected by a fingerprint access control system. If multiple modalities are used, then all modalities are required to be observed at the time of authentication, i.e., missing observations are usually not tolerated.

1.1 New Criteria

Compared to conventional systems, continuous verification imposes three important requirements on any multimodal fusion method (see Table 1).

It is clear that the usual fusion methods of sum, product etc., which are suitable for conventional systems cannot be directly used for continuous verification because they do not satisfy the above criteria. To be sure, some conventional systems do account for the reliability of different modalities, but few have to deal with missing modalities. Continuous verification systems, on the other hand, must operate *despite* missing modalities. Another difference is in the metrics used to measure performance. One-time verification systems typically use the False Accept and False Reject Rates (FAR and FRR, respectively) to quantify performance. These metrics are inadequate for continuous verification because *time* is not taken into account. For example, FAR and FRR do not measure whether the system responds quickly enough to stop an attack, nor how often the legitimate user is denied usage because of false rejects. These timing measurements are important yardsticks for continuous verification systems. Hence, new performance metrics are necessary, which we propose in this paper.

The key to continuous verification is the integration of biometric observations across both modalities and time. Fig. 1 shows an example of face and fingerprint observations being acquired over time. For simplicity, observations

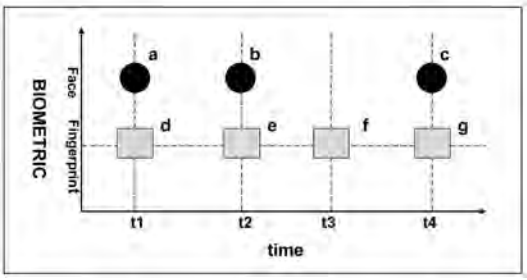

Fig. 1. Diagram illustrating how observations from different biometric modalities may occur over time.

from different modalities are shown to occur at the same time, but, in fact, our work here allows for asynchronous operation. Up to now, the task of integrating data across both modality and time has not been addressed satisfactorily. As Altinok and Turk state in [6, Section 2.2],

> Perhaps the best approach, but also the most complex to formulate, is to integrate in both directions (across modalities and across time) simultaneously, rather than sequentially.

In this paper, we present a holistic fusion method that combines face and fingerprint across modalities and time simultaneously and in a way that satisfies the above three criteria. This is realized by using a Hidden Markov Model (HMM) [7] in a Bayesian framework. Our method lends itself to an efficient recursive formulation and requires only one user-supplied parameter. This parameter determines how quickly old observations are discounted and can be tuned by the security administrator to fit the security policy of the site.

1.2 Related Work

To be sure, the idea of continuous verification/authentication is not new. Interest in this technology has been increasing in recent years, however. For example, Klosterman and Ganger [8] examined the differences between biometrics and traditional passwords/tokens and argued for the suitability of biometrics for continuous authentication. They enabled a Linux system to continuously authenticate its user via the Linux Pluggable Authentication Module (PAM) [9]. Because of high-computational cost, however, the actual authentication decision is made by a separate machine. Only a single modality, face, was used for authentication.

In response to the 11 September 2001 attacks, Carrillo proposed two designs that use continuous biometrics authentication to safeguard the aircraft cockpit against unauthorized control [10]. The designs differ in where the actual verification decision is made: either on board the aircraft or at distributed locations offsite. Carrillo's proposals allow for, and argue for, the use of multimodal biometrics. She also enunciated new procedures for biometrics authentication on the flight deck. However, Carrillo did not implement an actual system; hers was a design proposal only.

In [6], Altinok and Turk experimented with using voice, face, and fingerprint biometrics for continuous authentication. They articulated two key issues in continuous authentication, namely, the need for integration across both modality and time, and the requirement that the system must be able to determine "authentication certainty" at any point in time, even in the absence of observations. However, their work focused only on multimodal fusion and did not study the consequences of ephemeral misclassification on

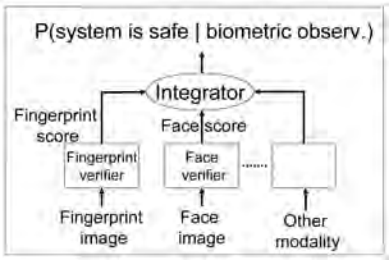

Fig. 2. The integration scheme of our system.

Fig. 3. Physical setup: A computer connected to a Canon VCC4 video camera and a SecureGen fingerprint mouse. The mouse has a fingerprint sensor to acquire images of the thumb.

system performance in a *real* system on realistic workloads. Moreover, their experiments used only simulated individuals because of the difficulty of getting real users.

As far as we can tell from the research literature, ours is the first implementation of a continuous verification system integrated into an operating system (OS) so that the OS can react to the presence/absence of the legitimate user. In our earlier implementation [11], we integrated a single modality (face) into the Linux 2.4.26 kernel [12]. We have since extended it to two modalities (face and fingerprint) [13], [14]. In this paper, we formalize the requirements for continuous verification and describe our usability test on real human subjects using our newly proposed performance metrics. For completeness, we also include details of our previous work cited in [13], [14].

1.3 Contributions

To summarize, our work makes the following contributions:

1. We articulate three important criteria for multimodal biometrics to be used in continuous verification. Continuous verification requires accounting for time *between* events rather than treating events as sequences. We account for this by decaying the last computed classification value. For this problem domain, it seems to be the natural thing to do; other application domains may model it differently.
2. We present a new and efficient Holistic Fusion method that satisfies these criteria.
3. We propose several new metrics to measure the performance of any continuous verification system: Time to Correct Reject, Probability of Time to Correct Reject, Usability, and the Usability-Security Curve. This is done because the traditional metrics of FRR and FAR are no longer appropriate for such systems.
4. We implement our continuous verification system into a computer operating system to make it reactive to user presence, thereby enabling the OS to protect interactive login sessions. It demonstrates that our ideas can be applied to real systems for use in regular computational activity. From a computer security perspective, it demonstrates a shift from *passive* to *active* security control.

An important application of biometrics is its use in authentication and this paper makes the case that previous techniques based on one-time verification *might not be adequate* for continuous verification. As personal computer systems become faster and accessories such as cameras and fingerprint readers become cheaper and part of standard computer configuration, we will see a broader use of continuous verification. This paper is a first step in that direction by proposing newer metrics against which to evaluate such systems and by suggesting a methodology by which an implementation can be realized in a useful, practical way.

1.4 Roadmap for the Paper

We explain the theory of our Holistic Fusion method, as well as the internals of our face and fingerprint verifiers in Section 2. We then briefly describe our implementation architecture and the OS kernel changes needed to make the system reactive to verification failures in Section 3. Next, we suggest three alternative fusion methods suitable for continuous verification in Section 4 and compare them with our Holistic Fusion method in Section 4.3. We also propose several new performance metrics for continuous verification systems and benchmark our work against these metrics in Section 5. Finally, we conclude our paper in Section 6 with a brief discussion on the computational overhead incurred by our system and some pointers to future work.

2 Biometrics and Their Fusion

The goal of verification is to determine whether the person with the claimed identity is who he claims to be. Two situations can occur: Either the verifier *accepts* the claim as genuine or the verifier *rejects* it (and decides that the user is an imposter).

In our case, the verification uses two types (modalities) of observations: fingerprint and face images. The challenge is to integrate these observations across modality and over time. To do this, we devised the integration scheme shown in Fig. 2. Currently, we implement a face verifier and a fingerprint verifier; other modalities are possible in the future. Each verifier computes a score (as well as the intra and interclass probabilities, see below) from its input biometric data, which is then integrated (fused) by the Integrator.

In general, fusion can be done at the feature, score, or decision level [2]. We choose to fuse at the score level because 1) this allows each verifier to operate asynchronously (which is not possible when fusing at the feature level), 2) additional modalities (e.g., voice) may be readily incorporated by simply adding a verifier for it, and 3) fusing at the decision level is too coarse (i.e., information is lost).

In the following sections, we describe in turn how we compute the score for each modality and how we fuse them into a single estimate.

We acquire fingerprint images using the SecureGen™ mouse (Fig. 3), which incorporates a fingerprint scanner ergonomically where the thumb would normally be placed. This makes the mouse a passive (nonintrusive) biometric

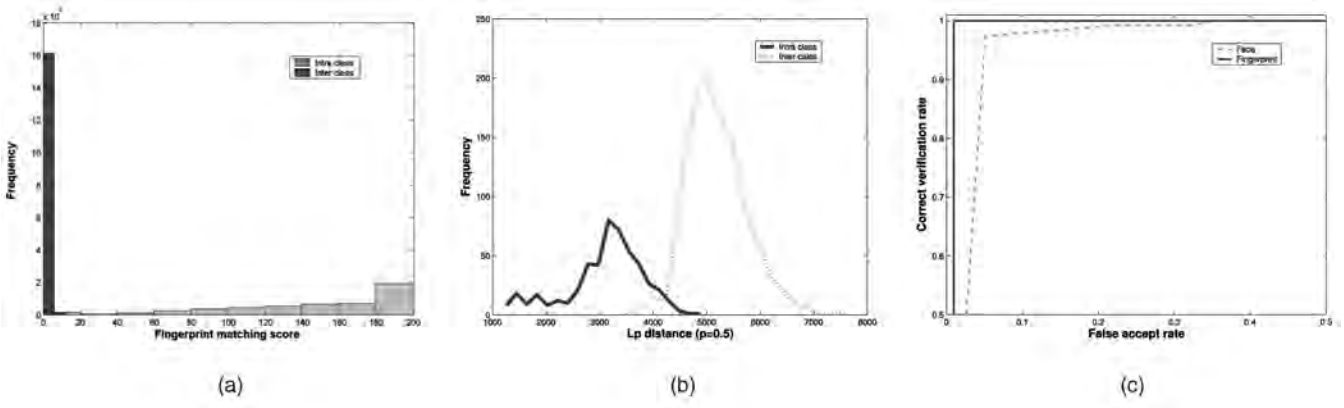

Fig. 4. (a) Fingerprint intraclass and interclass histograms for a typical user. These histograms do not overlap much, indicating that fingerprint verification is reliable. (b) Face intraclass and interclass histograms for the same user. There is greater overlap in these histograms than in fingerprint, indicating that face verification is less reliable than fingerprint verification. (c) ROC curves for Face and Fingerprint verifiers.

sensor, ideally suited for continuous verification. The mouse comes with a software development kit (SDK) that matches fingerprints, i.e., given two images, it computes a similarity score between 0 (very dissimilar) and 199 (identical). Unfortunately, the matching algorithm is proprietary and is not disclosed by the vendor. Nevertheless, this score is sufficient for our purpose. Besides the score, the algorithm also outputs another number between 0 and 100 that measures image quality.

We design our Fingerprint Verifier in two stages. In the training stage, we collect 1,000 training fingerprint images from each of several users. Then, for each user, we divide the training images into two sets: those belonging to the user (intraclass images, denoted by Ω_{intra}) and those belonging to others (interclass images, denoted by Ω_{inter}). For the set Ω_{intra}, we compute the pairwise image similarity using the proprietary algorithm and determine the histogram (or probability density function, pdf) of the resulting scores, s. That is, we determine $P(s \mid \Omega_{intra})$. Also, for each image in Ω_{inter}, we compute the its similarity with every image in Ω_{intra}. This produces the histogram of interclass scores $P(s \mid \Omega_{inter})$. Fig. 4a shows these pdfs for a typical user. Note that the pdfs do not overlap much, indicating that fingerprint verification is reliable (high verification accuracy). By studying these pdfs for different users, we discover that they are all very similar. We surmise that this is a feature of the proprietary algorithm. We therefore exploit this by assuming that all users have the same intra and interclass pdfs.

In the enrollment stage, we capture several fingerprint images of the user to be enrolled (who may not be in the training set) and retain one with the highest image quality. This will be used as the user's template for subsequent matching.

Now, given a new fingerprint image and a claimed identity, the image is matched against the claimed identity's template to produce a score s. From this, we compute $P(s \mid \Omega_{intra})$ and $P(s \mid \Omega_{inter})$ using the common intra and interclass pdfs learned during the training stage. If the fingerprint is indeed from the correct user, we expect $P(s \mid \Omega_{intra})$ to be high and $P(s \mid \Omega_{inter})$ to be low and vice versa. This is the classical Bayesian decision. However, we delay making this decision and instead pass these probability values to the Integrator to arrive at the overall decision. See Section 2.3 for more details.

2.1 Face Verifier

Our Face Verifier is also based on intra and interclass pdfs, except that the score s is now an image distance, rather than a similarity measure. As in the Fingerpint Verifier, we proceed in two stages. In the training stage, we capture 500 images of each user under varying head poses, using a Canon VCC4 video camera and the Viola-Jones face detector [15]. The images are resized to 28×35. For each user, we determine the intra and interclass pdfs as before (Section 2.1). However, this time the score is an image distance using the L_p norm (described below). This is similar to the ARENA method [16]. Fig. 4b shows these intra and interclass pdfs for one user. Compared to the fingerprint pdfs (Fig. 4a), there is greater overlap here. This indicates that the Face Verifier will not perform as well as the Fingerprint Verifier.

The L_p norm is defined as $L_p(\mathbf{a}) \equiv (\sum |a_i|^p)^{\frac{1}{p}}$, where the sum is taken over all pixels of image $\mathbf{a}$. Thus, the distance between images $\mathbf{u}$ and $\mathbf{v}$ is $L_p(\mathbf{u} - \mathbf{v})$. As in ARENA [16], we found that $p = 0.5$ works better than $p = 2$ (Euclidean).

Unlike the Fingerprint Verifier, however, we discover that different users have greatly different intra and interclass pdf pairs. It is therefore not possible to assume a common pair of pdfs for everyone. Thus, when enrolling a new user, we have to acquire and store several hundred images to form the user's Ω_{intra}. From this set, we also learn the user's intra and interclass pdfs.

Given a new face image and a claimed identity, we compute the smallest L_p distance between the image and the intraclass set of the claimed identity. This distance is used as a score s to compute $P(s \mid \Omega_{intra})$ and $P(s \mid \Omega_{inter})$, which are then used by the Integrator to determine an overall decision.

2.2 Holistic Fusion

At the heart of our technique is the integration of biometric observations across modalities and over time. This is done by the Integrator using an HMM (Fig. 5a) [7], which is a sequence of states x_t that "emit" observations z_t, for time $t = 1, 2, \ldots$. Each state can assume one of two values: $x_t \in \mathcal{S}$, where $\mathcal{S} = \{Safe, Attacked\}$. The state *Safe* means that the logged-in user is still present at the computer console, while the state *Attacked* means that an imposter has taken over control. It is also possible for the user to be absent from the console, but, for a high security environment, this is considered to be the same as *Attacked*. Each observation z_t

is either a face or fingerprint image or, equivalently, its corresponding score (see Sections 2.1 and 2.2). Note that the states are hidden (unobservable) and the goal is to infer the state from the observations.[2]

The result of the fusion is the calculation of P_{safe}, the probability that the system is still in the *Safe* state. This value can then be compared to a predefined threshold T_{safe} set by the security administrator,[3] below which appropriate action may be taken. A key feature of our method is that we can compute P_{safe} at any point in time ("authentication certainty"), whether or not there are biometric observations (Criterion 3). In the absence of observations, there is a built-in mechanism to decay P_{safe} reflecting the increasing uncertainty that the system is still *Safe* (Criterion 2).

Let $\mathcal{Z}_t = \{z_1, \ldots, z_t\}$ denote the history of observations up to time t. We assume that no two observations occur at *exactly* the same instant. This is usually the case if the hardware clock precision is high enough. Thus, each z_k is either a face or fingerprint observation. From a Bayesian perspective, we want to determine the state x_t that maximizes the posterior probability $P(x_t \mid \mathcal{Z}_t)$. Our decision is the greater of $P(x_i = \mathit{Safe} \mid \mathcal{Z}_t)$ and $P(x_t = \mathit{Attacked} \mid \mathcal{Z}_t)$. Equivalently, we may seek to determine if $P(x_t = \mathit{Safe} \mid \mathcal{Z}_t) > 0.5$ since the posterior probabilities must sum to 1. Using a little algebra, we may write:

$$P(x_t \mid \mathcal{Z}_t) \propto P(z_t \mid x_t, \mathcal{Z}_{t-1}) \cdot P(x_t \mid \mathcal{Z}_{t-1}), \tag{1}$$

$$P(x_t \mid \mathcal{Z}_{t-1}) = \sum_{x_{t-1} \in \mathcal{S}} P(x_t \mid x_{t-1}, \mathcal{Z}_{t-1}) \cdot P(x_{t-1} \mid \mathcal{Z}_{t-1}). \tag{2}$$

This is a recursive formulation that leads to efficient computations.[4] The base case is, of course, $P(x_0 = \mathit{Safe} \mid \mathcal{Z}_0) = 1$ because we know that the system is *Safe* immediately upon successful login. Observe that the state variable x_t has the effect of summarizing all previous observations. Because of our Markov assumptions, we note that $P(z_t \mid x_t, \mathcal{Z}_{t-1}) = P(z_t \mid x_t)$ and $P(x_t \mid x_{t-1}, \mathcal{Z}_{t-1}) = P(x_t \mid x_{t-1})$. Thus, we can rewrite (1) and (2) as

$$P(x_t \mid \mathcal{Z}_t) \propto P(z_t \mid x_t) \cdot P(x_t \mid \mathcal{Z}_{t-1}), \tag{3}$$

$$P(x_t \mid \mathcal{Z}_{t-1}) = \sum_{x_{t-1} \in \mathcal{S}} P(x_t \mid x_{t-1}) \cdot P(x_{t-1} \mid \mathcal{Z}_{t-1}). \tag{4}$$

However, $P(z_t \mid x_t)$ is simply the intraclass pdf (when $x_t = \mathit{Safe}$) or the interclass pdf (when $x_t = \mathit{Attacked}$). Note that, by using these pdfs, our fusion method implicitly factors in the reliability of each modality (see Fig. 4), thus satisfying Criterion 1. As for $P(x_t \mid x_{t-1})$, this is described by the state transition model shown in Fig. 5b. In the *Safe* state, the probability of staying put is p, while the probability of transitioning to *Attacked* is $(1-p)$. Once in the *Attacked* state, however, the system remains in that state and never transitions back to *Safe*.

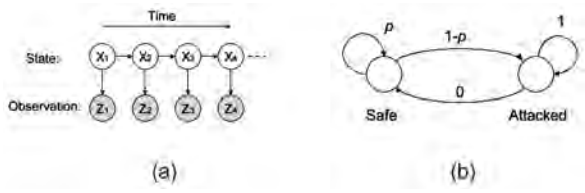

Fig. 5. (a) A Hidden Markov Model (HMM). (b) The state transition diagram.

The value of p is governed by domain knowledge—if there is no observation for a long period of time, we would like p to be small, indicating that we are less certain that the user is still safe (and, thus, more likely to have been attacked). To achieve this effect, we define $p = e^{k\Delta t}$, where Δt is the time interval between the current time and the last observation and k is a free parameter that controls the *rate of decay*, which the security administrator can define. For instance, if the security administrator decides that p should drop to half in 30 seconds, then $k = -(\log 2)/30$. In general, any function with a suitable rate of decay may be used to specify p. We chose an exponential function for its simplicity: A value of $k = 0$ means that the user is never attacked ($p = 1$), while a very large value of k indicates that attacks are very likely.

We further remark that our Holistic Fusion method is not the usual fusion rule typically used in combining classifiers. It is not a variant of the sum or product rules. Rather, it combines the likelihoods of each modality over time, weighted by the time interval between observations. This is best seen by expanding (3) and (4) for the case $t = 2$. After simplifying, we get the following:

$$\begin{aligned} P(x_2 = \mathit{Safe} \mid \mathcal{Z}_2) \propto\ & \overbrace{P(z_2 \mid x_2 = \mathit{Safe})}^{\text{intraclass}} \times \\ & e^{k\Delta t_2} \times \overbrace{P(z_1 \mid x_1 = \mathit{Safe})}^{\text{intraclass}} \times e^{k\Delta t_1}, \end{aligned} \tag{5}$$

$$\begin{aligned} P(x_2 = \mathit{Attacked} \mid \mathcal{Z}_2) \propto\ & \overbrace{P(z_2 \mid x_2 = \mathit{Attacked})}^{\text{interclass}} \times \\ & \Big[\big(1 - e^{k\Delta t_2}\big) \times \overbrace{P(z_1 \mid x_1 = \mathit{Safe})}^{\text{intraclass}} \times e^{k\Delta t_1} + \\ & \overbrace{P(z_1 \mid x_1 = \mathit{Attacked})}^{\text{interclass}} \times \big(1 - e^{k\Delta t_1}\big)\Big], \end{aligned} \tag{6}$$

where Δt_2 and Δt_1 are the time intervals between the second and first observations and between the first observation and the time of login, respectively. Terms of the form $e^{k\Delta t}$ come from the transition probabilities of the HMM.

As can be seen, the annotated forumulae above show how the fusion is accomplished in terms of the intra and interclass pdfs (likelihoods) of the different modalities and in terms of past observations weighted by the HMM transition probabilities. While such a formulation might not at first appear to be an intuitive way to fuse different modalities, we assure the reader that it is based on a

2. To be precise, our HMM differs from the conventional model in two ways. 1) Instead of the usual Decoding Problem, which asks for the most probable sequence of states given a history of observations (see [17]), we are asking for the most probable *current state* given the observations. 2) Our transition probabilities are not fixed, but rather a function of the elapsed time between observations.

3. According to Bayes' theory, T_{safe} must be 0.5. However, this may be too strict in practice. Security administrators generally prefer a flexible threshold that can be adjusted to suit their security policy. Moreover, several thresholds may be required in general, e.g., one to deny write-access, another to control read-access, and a third to grant access to operating system calls, etc.

4. At time t, if there exists a biometric observation, we use (1) to compute P_{safe}, otherwise (2).

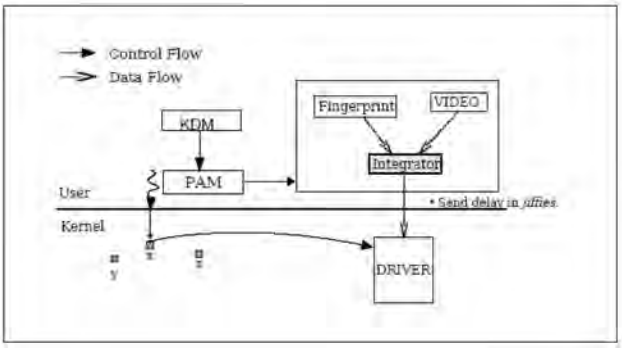

Fig. 6. Architecture of our continuous verification system integrated into the OS.

Bayesian approach of asking for the most probable (hidden) current state given the history of observations.

2.3 Summary

We encourage the reader to see the Appendix for a simple numerical example of our Holistic Fusion method in action. Meanwhile, we summarize our theory in a few key points.

1. We combine face and fingerprint modalities at the score level, using an integration scheme that can readily accommodate new modalities.
2. Our Holistic Fusion method implicitly incorporates the different reliability of each verifier by using the intraclass pdf (which is $P(z_t \mid x_t = Safe)$ in (1)) and interclass pdf (which is $P(z_t \mid x_t = Attacked)$). A verifier whose intra and interclass pdfs overlap little is more reliable than one whose pdfs overlap more. This satisfies Criterion 1.
3. The values for $P(x_t \mid x_{t-1})$ in (2) come from the transition probabilities of our Markov Model and these depend on p, which varies according to the time elapsed between observations. Thus, our Holistic Fusion method discounts older observations. This satisfies Criterion 2.
4. If an observation is available, we use (1) to determine P_{safe} . Otherwise, we use (2). Thus, we can determine "authentication certainty" at any time. This satisfies Criterion 3.
5. Because our hardware clock precision is one millisecond, it is very unlikely that observations from both modalities will occur *exactly* simultaneously. Therefore, each observation z_k is from either a face or fingerprint image.
6. Our Holistic Fusion is *not* a variant of the usual sum, product, etc. rules, nor is it a variant of the alternative fusion methods that we propose in Section 5. Rather, it is derived from a Bayesian perspective, and combines different modalities as they occur over time.[5]

3 Architecture

Fig. 6 depicts the various elements of our implementation and how they are integrated into the operating system. We have implemented this architecture on the Linux 2.4.26 kernel [12] with the KDE graphical environment running on the Redhat 9.0 distribution. For face image capture, we use the Euresys Picolo capture card and the Canon VCC4 camera. The captured images have a resolution of 768×576 pixels and are 24-bit deep. The fingerprint images are captured using the Secugen OptiMouse III. All experiments were performed on an Intel Pentium 2.4 Ghz machine with 512 MB RAM. In the interest of space, we will only briefly describe our architecture here. For more details, please refer to our earlier paper [13].

When a user logs in at the console using the kdm session manager [18], kdm authenticates the user using a password.[6] Additionally, it starts the face and fingerprint verifiers and initializes the *integrator* with the user-id of the user that has logged in.

Once the integrator has the user-id of the logged in user, it loads the biometric *profile* (the biometric features to be used for verification; in our case, the intra and interclass pdfs (Section 2)) corresponding to the user and starts acquiring biometric observations using the video and fingerprint boxes in Fig. 6. The arrows in the diagram denote the direction of data flow. The integrator is the central coordinating entity in the architecture that performs the following tasks:

1. It combines the verification results from different modalities obtained at different times into P_{safe}, the probability that the computer system is still *Safe* (Section 2.3).
2. It periodically communicates P_{safe} (indirectly, it actually computes and communicates a delay value, in jiffies) to the kernel so that the kernel can appropriately freeze or delay processes.

The driver box in the kernel space is responsible for controlling the processes owned by the user. It can delay or freeze processes based on the P_{safe} received from the integrator. This is achieved by adding a callback to where system calls are dispatched. When a user process makes a system call, it traps into the OS kernel and executes the callback routine in the driver. The driver now has control of that process and can decide its fate depending on P_{safe}.

3.1 Windows XP Implementation

We have also successfully implemented our continuous verification ideas into the Windows XP Professional™ operating system. Because of disclosure restrictions, we can only describe the implementation briefly. Analogous to the integrator, an OS process periodically acquires biometrics observations and calculates the fused probability P_{safe}. Kernel calls are then used to suspend or resume user processes accordingly. In addition, there is a filtering mechanism to prevent events such as keyboard or mouse clicks from being sent to the suspended processes. In terms of computational overhead, our Windows implementation suffers a similar degradation: about 25 percent increase in elapsed time.

5. We note that the usual fusion rules can also be given a Bayesian interpretation. See, for example, [3].

6. Verification at login time need not require biometrics; other forms of authentication, say, passwords or tokens, may be preferred. Our concern in this paper is with continuous verification rather than with one-time verification during login. Thus, in our implementation, we simply use passwords.

4 Alternative Fusion Methods

We note that our Holistic Fusion method is not the only one that satisfies the three criteria of Table 1. Other fusion methods are possible. At this juncture, it is instructive to consider alternative fusion methods and to compare them. We describe three: Temporal-first, Modality-first, and Naive Integration. Temporal-first is the approach that combines temporal information first, then integrates over different modalities. (Altinok and Turk [6] used a similar scheme.) On the other hand, Modality-first integrates modality information first, then integrates across time. The idea of Naive Integration is that we always use the most accurate modality that is available; otherwise, in the absence of observation, we simply decay P_{safe}. In the following sections, we provide more details. We emphasize again that our Holistic Fusion is different from these alternative methods: It is markedly different and is not a variant of these methods.

4.1 Temporal-First and Modality-First Integration

Fig. 1 shows how observations from different modalities present themselves over time. Observations from a single modality are shown horizontally, while observations across modalities are shown vertically. Note that, at time t_3, only fingerprint is observed. Also, note that, for ease of understanding, we show observations a and d as aligned vertically. In practice, we allow a and d to occur within a small window of time apart.

One common method of fusion is the sum rule, which has been shown to be robust to estimation errors and thus to perform more reliably than the product, median, min, max, and majority vote rules (see [3]). However, the usual formulation of the sum rule cannot be directly applied here because it does not satisfy the Three Criteria for continuous verification (Table 1). But, it is possible to adapt it as follows: Let $P(x_t \mid \mathcal{Z}_t^{m_j})$ denote the posterior probability of being safe at time t for modality m_j. To combine across time, compute the weighted sum:

$$P\left(x_t \mid \mathcal{Z}_t^{m_j}\right) = \frac{1}{N}\sum p\left(x_{t_i} \mid z_{t_i}^{m_j}\right) \cdot e^{k\Delta t_i}, \tag{7}$$

where Δt is the time difference between the current time and observation time, N is the number of observations. This decays older observations by $e^{k\Delta t}$ and, thus, satisfies Criterion 2.

To combine over modalities, we may again use a weighted sum:

$$P(x_{t_i} \mid z_{t_i}) = w^{m_1} \cdot P(x_t \mid z_{t_i}^{m_1}) + w^{m_2} \cdot P(x_t \mid z_{t_i}^{m_2}). \tag{8}$$

Note that here the two weights are w^{m_1} and w^{m_2}. They should sum to unity and be chosen to reflect the reliability of each modality in order to satisfy Criterion 1. In our work, we use the area under the ROC curve (see Fig. 4c) to represent the reliability.

Thus, Temporal-first implies the application of (7) followed by (8). Similarly, Modality-first changes this construction by applying (8) first, then (7). Note that if there is only a single modality (i.e., time t_3 in Fig. 1), we just use the modality (no weight applied) as the combined result. Likewise, if there is only one observation across time, then we just decay the observation by $e^{k\Delta t}$. In practice, we combine observations within a recent history H from the current time since observations that are too old have weights close to zero.

4.2 Naive Integration

From Fig. 4c, we see that fingerprint is more reliable than face (at least for our data set). The idea of Naive Integration is to use the most reliable modality available at any time instant. More precisely:

1. At any time t, if a fingerprint observation exists, then $P(x_t \mid \mathcal{Z}_t) = P(x_t \mid z_t^{m_2})$ $(m_2 = fingerprint)$ whether or not face observation exists.
2. Otherwise, if there exists only face observation, then $P(x_t \mid \mathcal{Z}_t) = P(x_t \mid z_t^{m_1})$ $(m_1 = face)$ since now face is the most reliable biometric that is available.
3. Else, if no biometric observation is available, then we just decay the probability $P(x_t \mid \mathcal{Z}_t) = P(x_{t-1} \mid z_{t-1}) \cdot e^{k\Delta t}$, where $P(x_{t-1} \mid z_{t-1})$ is calculated from Steps 1 or 2, depending on the last biometric observation (fingerprint or face). Here, Δt is the time interval between the current instant and the last observation.

It is clear that Naive Integration satisfies the Three Criteria in Table 1. Note also that Naive Integration is similar to the max rule in that it selects the maximum probability. However, Naive Integration chooses the a posteriori probability of the *most reliable* verifier rather than the maximum of the probabilities themselves.

4.3 Experiments

How do these fusion methods compare? We run a number of experiments on real users to measure their performance. We are interested in the following questions: 1) How do these methods compare when the legitimate user is using the system? Are there frequent rejects? 2) How quickly do these fusion methods detect an imposter attack? Intuitively, the faster the detection, the less opportunity for the imposter to inflict damage. 3) What happens when there is partial impersonation, i.e., when only one modality is being impersonated? How reliably can the fusion method detect such a situation?

First, we determine the performance of each individual verifier. Fig. 4c shows the Receiver-Operating-Characteristic (ROC) curves for both the fingerprint and face verifiers. These curves are determined as follows: Each verifier is given an image X to authenticate against the claimed identity D. The image X is equally likely to be from the legitimate user or an imposter. Using the maximum-likelihood ratio [17], the verifier computes $P(X|D) - P(X|\neg D) > \sigma$, where σ is a threshold specific for the verifier. If true, the verifier accepts the image and, if false, the verifier rejects it. The False Accept and False Reject Rates (FAR and FRR) can then be calculated. The ROC curve is obtained by plotting $1 - FRR$ versus FAR as σ is varied.

The area under the ROC curve is a measure of the reliability of the verifier. The ROC areas for fingerprint-only, and face-only verifiers are 0.9995 and 0.970, respectively, thus showing that fingerprint is more reliable than face. This seems to suggest that using fingerprint alone is sufficient for continuous verification. However, as we pointed out in the Three Criteria, combining multiple biometrics is preferred over using just a single modality. The lack of observations from a single modality can be compensated for by using a second modality. Also, it is more difficult for an imposter to impersonate multiple biometrics simultaneously.

4.4 Comparing the Fusion Methods

We run four experiments to evaluate how the system behaves when one or both of the biometrics are impersonated. In these, we take turns to impersonate each modality at a time. Because each user presents his biometric in a different way, we cannot average the curves from different users. Figs. 7a, 7b, 8a, and 8b show five plots each in the following order: individual probabilities, Holistic Fusion, Naive Integration, Modality-first, and Temporal-first Integration. In these experiments, $\Delta t = 1.5s$ is used for modality integration, $H = 30s$ for temporal integration and $k = -log(2)/30$ for the decay function. There can be no observations at some time periods. In these situations, we disable the system in order to maintain its integrity. The user has to relogin to regain access. These four setups can be classified into three cases:

1. **Legitimate user using the system**. Fig. 7a shows the biometric observations for 15 minutes. The individual probabilities P_{safe} (Fig. 7a(1)) are not consistently high; they occur in a sporadic manner. This means that any value for the threshold T_{safe} will result in significant False Accept and False Reject rates. In continuous verification, a False Accept is a security breach, while a False Reject inconveniences the legitimate user because he must reauthenticate himself. Ideally, P_{safe} should not fluctuate, but be equal to 1 as long as observations are available. Of the four fusion methods, Holistic Fusion comes closest to this ideal (Fig. 7a(2)). It computes a P_{safe} value close to 1, except for periods in which there are no observations from both modalities (around 300s and 600s). At such time, P_{safe} decreases gradually according to the decay function. By comparison, the P_{safe} computed by Naive Integration (Fig. 7a(3)) fluctuates wildly because only a single modality is used any at time. Again, this means no T_{safe} value will make both FRR and FAR small. As for Modality-first (Fig. 7a(4)) and Temporal-first (Fig. 7a(5)) Integration, the plots are similar. The P_{safe} values are not close to 1. Moreover, in the absence of observations, P_{safe} drops abruptly to zero, resulting in sudden lock outs. From these plots, it is clear that Holistic Fusion is superior to the other fusion methods.
2. **Imposter taking over the system**. Fig. 7b shows the observations when an imposter takes over the system at some time instant (at around 38s). The probabilities of individual biometrics (Fig. 7b(1)) as well as P_{safe} for all integration methods drop to near zero after the attack. The goal here is to detect the attack as soon as possible so that damage to the system is minimized. Both Holistic Fusion (Fig. 7b(2)) and Naive Integration (Fig. 7b(3)) detect this situation sooner than the other two methods. However, P_{safe} for Naive Integration does not remain consistently low; it fluctuates widely. This implies that $FAR > 0$ for most values of T_{safe}. For Modality-first (Fig. 7b(4)) and Temporal-first (Fig. 7b(5)) Integration, the system takes longer to detect the imposter (when $T_{safe} = 0.5$). Choosing a larger value for T_{safe} can reduce the time to detection, but at the expense of a higher FRR. The best method is Holistic Fusion, which detects the imposter quickly (within 3s in our experiments) and whose P_{safe} remains low after the attack.

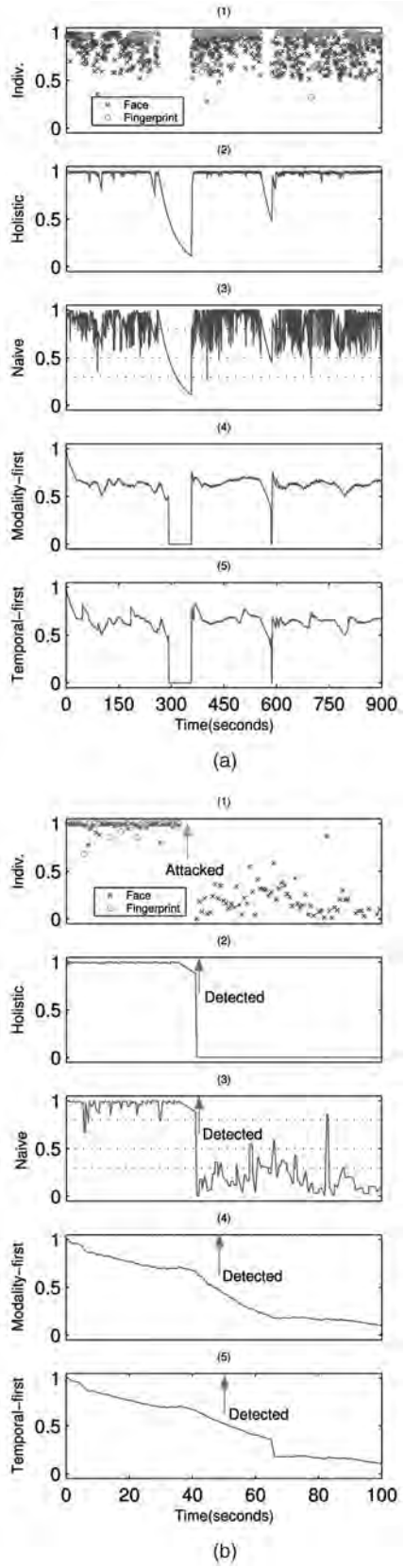

Fig. 7. (a) Legitimate user using the system for 15 minutes. (b) Imposter taking over the system.

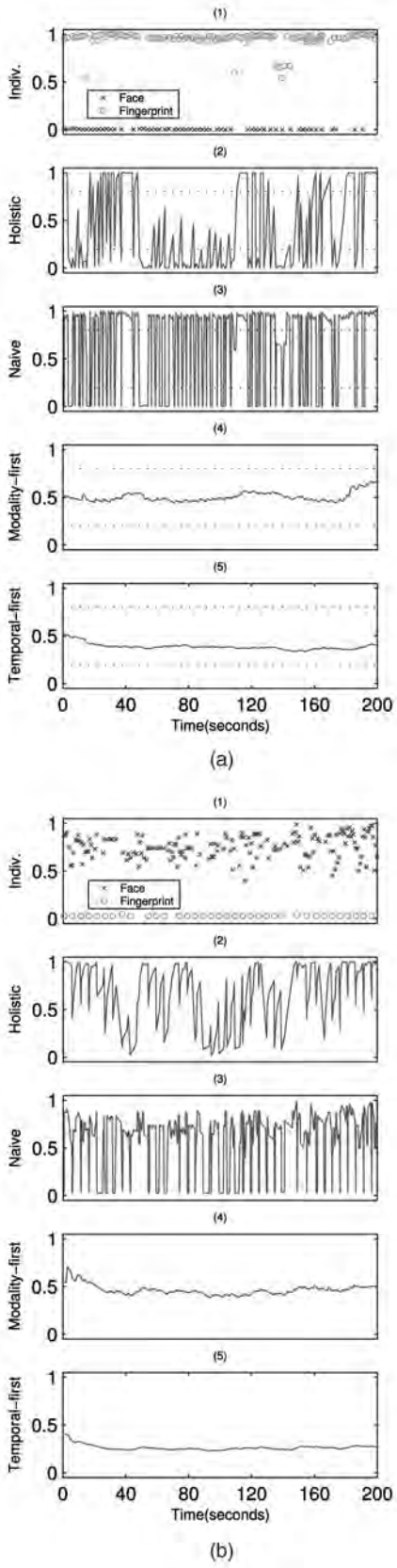

Fig. 8. Partial impersonation. (a) Genuine fingerprint + Fake face. (b) Fake fingerprint + Genuine face.

3. **Imposter successful in faking one of the biometric (Partial impersonation)**. Fig. 8a(1) depicts a situation where the imposter has successfully faked the fingerprint but not face. The individual probabilities contradict each other, and result in wildly fluctuating plots in both Holistic Fusion (Fig. 8a(2)) and Naive Integration (Fig. 8a(3)). This gives us a way to detect partial impersonation: We may just take two thresholds, one high and one low (say, 0.8 and 0.2) and simply count the number of times within a fixed time interval that P_{safe} jumps between these thresholds. However, comparing Fig. 8a(3) and Fig. 7a(3), we see that Naive Integration cannot distinguish between partial impersonation and the legitimate user. Fluctuating P_{safe} values seem to be an inherent property of Naive Integration. The plots for Modality-first (Fig. 8a(4)) and Temporal-first (Fig. 8a(5)) Integration are relatively flat and are in fact similar to those in Fig. 7a(4) and Fig. 7a(5) (except when there are completely no biometric observations). Again, this means these two methods cannot distinguish partial impersonation from legitimate usage. Only Holistic Fusion provides a way to detect partial impersonation that is different from detecting the real user.

We remark that this fluctuating behavior of Holistic Fusion may be intuited from examining (3) and (4). By expanding and simplifying these equations, we can get:

$$P(x_t = Safe \mid \mathcal{Z}_t) \propto P(z_t \mid x_t = Safe) \times p \times P(x_{t-1} = Safe \mid \mathcal{Z}_{t-1}).$$

In other words, P_{safe} at time t is proportional to the product of the intraclass pdf, the HMM transition probability, and P_{safe} at time $t-1$. Suppose the previous P_{safe} is high (say, from a succesfully faked fingerprint) and the current observation is the imposter's face, then $P(z_t \mid x_t = Safe)$ will be low, indicating a poor face match. In turn, this results in a lower value for the current P_{safe} (assuming the transition probability p remains constant). Conversely, a previously low P_{safe} will be "pulled up" by a high likelihood coming from a good biometric match. The alternating (and conflicting) modalities give rise to fluctuating P_{safe} values.

What happens if an imposter is careful not to present any observation (neither face nor fingerprint)? In this case, P_{safe} decreases to zero due to the decay function. This is also the situation if the legitimate user has left the console without logging off. In either case, system integrity is ensured.

5 Performance Metrics

We now turn our attention to performance. As we argued in Section 1, the traditional metrics of FAR and FRR are no longer appropriate to characterize the performance of continuous verification systems because *time* is not accounted for in these metrics. We thus begin this section with definitions of new performance metrics. Then, we benchmark our continuous verification system against these new metrics. We hope that our metrics will be adopted by other researchers to measure the performance of *any* continuous verification system.

5.1 Definitions

As in one-time verification systems, we need different metrics depending on whether the user is legitimate or an imposter. We then need a metric that succinctly captures the overall performance of the system. This will allow for different systems to be compared.

1. **Time to Correct Reject** (TCR). If the user is an imposter, we are concerned with how quickly the continuous verification system detects this situation. We define TCR to be *the interval between the start of the first action taken by the imposter to the time instant that the system decides to (correctly) reject him*. What constitutes an action by the imposter? This depends on the application. In our case of a desktop PC, we define an action to be either a keyboard or mouse click. In the case of an aircraft cockpit, this might be a movement of the joystick or an activation of some instrument on the control panel. We argue that TCR must be measured from the instant the imposter takes an action, rather than from his mere presence. If the imposter takes no action, then he is just an observer and cannot damage the protected system. There is no need to reject him in this scenario. We also note that there could be many false accepts before the system finally correctly rejects the imposter. TCR measures the time *until the first correct reject decision*. If the system fails to correctly reject the imposter, we define $TCR = \infty$. TCR is measured in seconds.

   Ideally, TCR should be zero; in reality, it is sufficient for TCR to be less than some *window of vulnerability*, W. This is the minimum time required for the imposter to damage the protected system. This time is clearly application-dependent. In our case, we set $W = 3$ seconds, which we empirically measured to be the time taken to type `rm -rf *` on our Linux console. As long as the system reacts faster than the imposter can do damage, system integrity is assured.
2. **Probability of Time to Correct Reject** ($PTCR$). Strictly speaking, requiring $TCR < W$ is enough to guarantee system integrity. But, a more nuanced metric may be more useful. We define the Probability of Time to Correct Reject, $PTCR$, as *the probability that* TCR *is less than* W: $Prb(TCR < W)$. Ideally, $PTCR$ should equal 1; but, for some applications, it may be tolerable for $PTCR$ to be < 1. This means that the system can sometimes take longer than W seconds to correctly reject an imposter. If the system always fails to correctly reject, then $PTCR = 0$ for all W. We remark that $PTCR$ is the analog of FAR.
3. **Usability**. For the legitimate user, we define the *Usability* of the system as *the fraction of the total time that the user is granted access to the protected resource*. As an example, suppose the legitimate user logs in for a total duration of T seconds. During this time, the system sometimes rejects the user (and, thus, denies the user access to the protected resource) and sometimes accepts him. Let t be the total duration that the system accepts the user. Then, $Usability = t/T$. $Usability$ is analogous to FRR.

   Ideally, $Usability = 1$, i.e., the legitimate user is granted access all the time. Any denial of access represents an inconvenience to the legitimate user because he must either reauthenticate himself or take other action to regain access. We could thus define $Inconvience$ as $1 - Usability$. We note that, in some cases, a user may not be completely denied access; the system may choose to give the user partial access instead. For instance, in our desktop PC application, it is conceivable for the system to delay user processes, rather than suspend them, when it decides to reject the user. Nevertheless, we consider this to be inconvenient because the delayed process is not executing at full speed. We also note that the notion of $Usability$ could depend on the type of activity performed. For instance, the system could deny write-access but still grant read-access to the user or permit Web browsing, but not composing a document. In such situations, it may be useful to also define an $Average\ Usability$ as the sum of activity-specific usabilities weighted by the fraction of time spent on each activity.
4. **Usability-Security Characteristic Curve** (USC). We can plot $Usability$ versus $PTCR$. Typically, a continuous verification system will make its accept/reject decision based on some threshold θ. By varying θ, the plot of $Usability$ versus $PTCR$ will trace the *Usability-Security Characteristic* curve. Fig. 10c shows some USC curves of our system. This is analogous to the ROC curve used to measure one-time verification systems. The area under the USC curve is therefore a measure of the overall performance of a continuous verification system.

With these new performance metrics, we can now benchmark our implementation against them. We describe this in the next section.

5.2 Benchmark

To evaluate the performance of our continuous verification system, we asked real people to use our system. To determine $Usability$, we had 11 legitimate users, all graduate students or research staff in our lab, perform a 30-minute long sequence of predefined activities: Web surfing (duration: 8 minutes), e-mail (duration: 7 minutes), entering text into an MS Word document (duration: 7 minutes), and watching a video (duration: 8 minutes). This mix of activities is to simulate a typical workload on a desktop computer. To determine $PTCR$, we got the two users to each impersonate 11 other people, for a total of 22 instances of impersonation. The imposter was asked to click on a message box, or to press RETURN, to begin the "Attack." This was recorded as the first action taken by the imposter.

Fig. 9 shows the biometric observations for one legitimate user over the 30-minute period. (It does not make sense to average the observations over all users, so we show only one.) We show both the individual modality probabilities, as well as the fused probability from our Holistic Fusion method. Fig. 10 shows the $Usability$ curves, the $PTCR$ curves, and the USC curves, respectively. These curves are the average of all user curves. We can make the following observations:

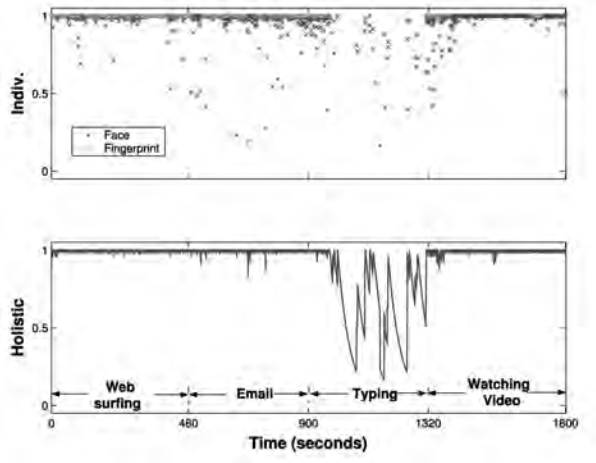

Fig. 9. Biometric observations for a legitimate user performing different activities.

1. The type of activities affect the presentation of biometrics (see the top plot in Fig. 9). Web surfing and e-mail require the user to face the camera and hold the mouse, resulting in observations for both modalities. By comparison, typing a document results in an absence of fingerprint observations because both hands are operating the keyboard rather than the mouse. Also, face observations are less frequent and fluctuate more. This is because the user needs to turn away from the camera to look at the text that he is typing. There is thus a greater incidence of nonfrontal head pose, which the face detector fails to detect. Watching videos is largely a face-only activity: There is little fingerprint observation because the user seldom uses the mouse.
2. From the output of our Holistic Fusion method (bottom plot of Fig. 9), we observe that P_{safe} is almost always 1 for Web surfing and video watching. This is because biometric observations from at least one modality are constantly available. P_{safe} fluctuates a little for e-mail because the user is sometimes looking away from the camera and his thumb is not on the mouse. P_{safe} fluctuates the most when the user is typing a document, that is, the user is frequently denied access for any reasonable decision threshold θ. This suggests that face and fingerprint modalities are not sufficient for continuous verification for such activities.
3. Fig. 10a is a plot of $Usability$ versus the threshold θ for the various activities. In our implementation, $\theta = T_{safe}$ and the system accepts if $P_{safe} > T_{safe}$ and rejects otherwise. A threshold of 0 means the system always accepts and never rejects so that $Usability = 1$, while a threshold of 1 means the system always rejects and, hence, $Usability = 0$. In between these values, we observe that the curves are relatively flat for all activities except typing a document, which seem to decrease linearly with increasing threshold (bottom curve). These curves can be used to determine a suitable threshold for all the activities (except typing).

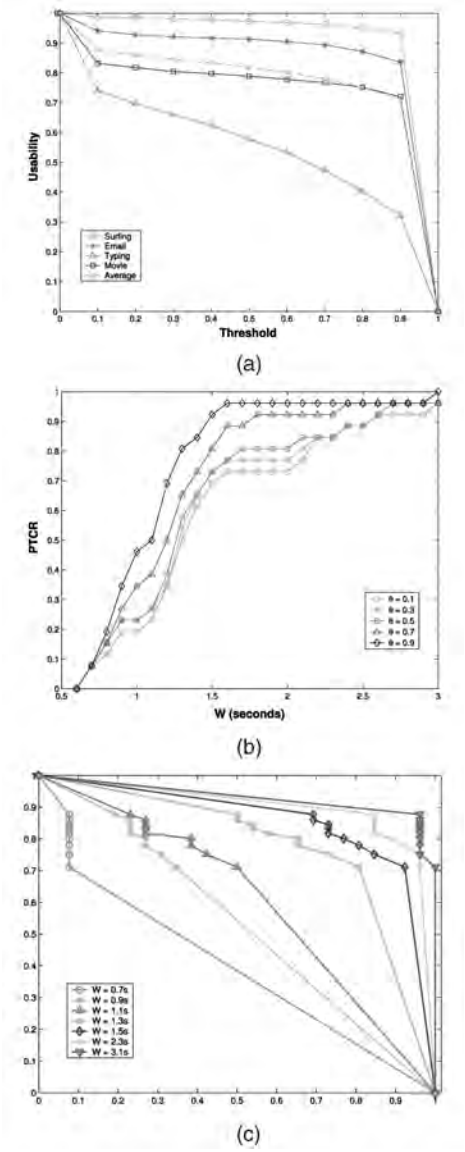

Fig. 10. (a) $Usability$ curves: plot of $Usability$ versus threshold for different activities. (b) $PTCR$ curves: plot of $PTCR$ versus W for different thresholds. (c) $Usability$-Security Characteristic curves: plot of $Usability$ versus $PTCR$ for various W.

We also plot an average usability curve (dotted), which is the weighted sum of the other curves. The weights are the fraction of time spent on each activity.

4. Fig. 10b is a plot of $PTCR$ versus the vulnerability window W, for various thresholds θ. Intuitively, a larger W means the system has more time to react to an imposter and, thus, $PTCR$ should be higher. This is borne out by the curves. Also, a larger threshold θ means a more conservative system (the system is more readily rejecting the user) and this explains why curves for larger θ values are on top of curves for smaller θ. These $PTCR$ curves may be used to determine a suitable choice of W. For $W > 1.5$ seconds, all curves flatten out, indicating that not much security is gained by increasing W beyond this point.
5. Fig. 10c shows the USC curves for different values of W. Recall that these curves are generated by varying θ.

If the system always accepts and never rejects, then $PTCR = 0$ for any W, and $Usability = 1$. If the system always rejects, then we can say that $TCR = 0$ and, hence, $PTCR = 1$ for any W. It also follows that $Usability = 0$. This explains the two extreme points of the curves. As for the shape of these curves, we observe that there is a trade-off between $Usability$ and $PTCR$. Greater usability comes at the expense of less security (increased vulnerability to attacks). This is analogous to the trade-off between FAR and FRR in the ROC curve. In general, we expect that a larger W value means more time for the system to react, for the same threshold θ. Thus, curves for larger W values lie to the right of curves for smaller W.

6 Conclusion

We began this research project because we observed that current one-time authentication systems are inadequate for high-security environments. Moreover, biometrics authentication has advanced rapidly in recent years so that what was previously computationally expensive and inaccurate is today feasible. Hence, this idea to use biometrics for continuous verification.

Our initial concern was overhead: How much more CPU time was needed to handle continuous verification? We performed a number of micro and macro benchmarks on system performance with and without continuous verification (details in [13]) and found that CPU-intensive tasks, such as program compilation, took 25 percent more time when continuous verification was turned on. The bulk of the time was taken by the face detector to locate faces in an image. This was not encouraging as we felt that 25 percent was rather high. However, when we put users to ordinary office tasks such as browsing the Web, reading documents, etc., we found that users did not perceive any performance degradation. This was probably because such tasks are more I/O intensive than CPU-intensive. The CPU was otherwise idle anyway, when waiting for the hard disk to fetch data, or for the user to type something. This was encouraging news.

Our next concern was whether such a system has ever been attempted before. To our surprise, there is no implementation reported in the research literature. The closest work to ours was that of Altinok and Turk [6]. There, the authors enunciated the need for fusion across modalities and time and the requirement for authentication certainty at all times. We adopted these ideas and formalized them into the Three Criteria for continuous verification.

This immediately called into question the applicability of standard classifier fusion methods. We found that standard methods were indeed inadequate and developed our own Holistic Fusion technique. As a comparison, we also adapted the sum rule to obtain the Temporal-first and Modality-first fusion methods, as well as devised the Naive Integration method. Our experiments provide evidence that Holistic Fusion is the best among these four.

We also discovered that the traditional performance metrics for one-time verification systems are no longer adequate. We thus proposed new metrics that incorporated *time* and which we hope other researchers will adopt.

Of course, the real crux is in an actual usability test, which we performed with 11 users over 30 minutes. This is by no means an extensive test, but even here, we observed that $Usability$ is generally high for various activities and that imposter attacks are detected well within 3 seconds. We are pleased with these results.

In the near future, we plan to conduct a larger usability test (more users, longer period of time). We also intend to investigate the use of keystroke dynamics as a possible biometric. This will no doubt help to increase usability in keyboard-intensive activities, such as typing a document. As for overhead, although users currently do not perceive any significant slowdown when performing ordinary tasks, there is still room for improvement. We plan to off-load verification/fusion computations to a secondary CPU, e.g., an FPGA processor.

Appendix

Numerical Example

In this section, we provide a simple numerical example to illustrate our Holistic Fusion. Consider the following scenario: The user logs in at time $t = 0s$ and begins using the system. At time $t = 0.3s$, the system acquires a face image with a score (L_p distance, see Section 2.2) of $z_1 = 4,000$. Then, at time $t = 0.9s$, the system acquires a fingerprint observation with a score of $z_2 = 175$. We now wish to calculate P_{safe} at the said times. In addition, we wish to know P_{safe} at a new time, $t = 1.3s$, at which there is no observation from either modality. For this example, we suppose that the user-defined parameter k is set to $k = -log(2)/30$.

Case A. Time $t = 0.3s$. Since there is only one observation so far, $\mathcal{Z}_0 = \emptyset$ (empty set) and the history of observations is $\mathcal{Z}_1 = \{z_1\}$. Based on (3) and (4), we have:

$$P(x_1 \mid \mathcal{Z}_1) \propto P(z_1 \mid x_1) P(x_1 \mid \mathcal{Z}_0), \tag{9}$$

$$P(x_1 \mid \mathcal{Z}_0) = \sum_{x_0 \in \mathcal{S}} P(x_1 \mid x_0) P(x_0 \mid \mathcal{Z}_0). \tag{10}$$

Substituting (10) into (9):

$$\begin{aligned} P(x_1 \mid \mathcal{Z}_1) &\propto P(z_1 \mid x_1) \times \\ &\quad \sum_{x_0 \in \mathcal{S}} P(x_1 \mid x_0) P(x_0 \mid \mathcal{Z}_0) \\ &= P(z_1 \mid x_1) \times \\ &[P(x_1 \mid x_0 = \mathit{Safe}) P(x_0 = Safe \mid \mathcal{Z}_0) + \\ &\quad P(x_1 \mid x_0 = \mathit{Attacked}) P(x_0 = \mathit{Attacked} \mid \mathcal{Z}_0)] \end{aligned} \tag{11}$$

$$= P(z_1 \mid x_1) P(x_1 \mid x_0 = \mathit{Safe}), \tag{12}$$

where we used the fact that $P(x_0 = \mathit{Attacked} \mid \mathcal{Z}_0) = 0$ and $P(x_0 = Safe \mid \mathcal{Z}_0) = 1$.

From Fig. 5b, the transition probabilities are: $P(x_1 = \mathit{Safe} \mid x_0 = \mathit{Safe}) = p$ and $P(x_1 = \mathit{Attacked} \mid x_0 = \mathit{Safe}) = 1 - p$. Here, $p = e^{k\Delta t}$, where $\Delta t = 0.3$. Substituting this into

(12), we can obtain the probability of being safe and the probability of being attacked, respectively.

$$P(x_1 = Safe \mid \mathcal{Z}_1) \propto P(z_1 \mid x_1 = Safe)e^{k\Delta t}$$
$$P(x_1 = Attacked \mid \mathcal{Z}_1) \propto P(z_1 \mid x_1 = Attacked) \times (1 - e^{k\Delta t}).$$

The proportionality in the above equations may be resolved by noting that the left-hand sides must sum to 1. Hence,

$$P(x_1 = Safe \mid \mathcal{Z}_1) = \frac{P(z_1 \mid x_1 = Safe)e^{k\Delta t}}{P(z_1 \mid x_1 = Safe)e^{k\Delta t} + P(z_1 \mid x_1 = Attacked)(1 - e^{k\Delta t})}.$$

To obtain $P(z_1 = 4,000 \mid x_1 = Safe)$, we note that this is nothing but using the face intraclass pdf to compute the probability of the score. Likewise, $P(z_1 = 4,000 \mid x_1 = Attacked)$ is computed from the interclass pdf. From Fig. 4b, we see that $P(z_1 = 4,000 \mid \Omega_{intra}) = 20$ and $P(z_1 = 4,000 \mid \Omega_{inter}) = 10$. Plugging in all the numerical values, we arrive at:

$$\text{At } t = 0.3s, P_{safe} = P(x_1 = Safe \mid \mathcal{Z}_1) = 0.9965. \quad (13)$$

Case B. Time $t = 0.9s$. Now, the observation history is $\mathcal{Z}_2 = \{z_2 = 175, z_1 = 4,000\}$. Similarly to (11), we may derive the following:

$$P(x_2 \mid \mathcal{Z}_2) \propto P(z_2 \mid x_2) \times [P(x_2 \mid x_1 = Safe)P(x_1 = Safe \mid \mathcal{Z}_1) + P(x_2 \mid x_1 = Attacked)P(x_1 = Attacked \mid \mathcal{Z}_1)]. \quad (14)$$

Note that the quantities $P(x_1 = Safe \mid \mathcal{Z}_1)$ and $P(x_1 = Attacked \mid \mathcal{Z}_1)$ may be obtained from (13), while the values of $P(x_2 \mid x_1)$ are the transition probabilities from the HMM. However, the time interval is now $\Delta t = 0.9 - 0.3 = 0.6$. As before, the values of $P(z_2 = 175 \mid x_2 = Safe)$ and $P(z_2 = 175 \mid x_2 = Attacked)$ are obtained using the intra and interclass pdfs. From the fingerprint pdfs of Fig. 4a, these are 125 and 5, respectively. Hence,

$$P(x_2 = Safe \mid \mathcal{Z}_2) \propto 125 \times \{0.0035 \times 0 + 0.9965 \times 0.9862\} = 122.8435,$$
$$P(x_2 = Attacked \mid \mathcal{Z}_2) \propto 5 \times \{0.0035 \times 1 + 0.9965 \times (1 - 0.9862)\} = 0.0863.$$

Finally, normalizing the above equations so that the left-hand sides sum to unity gives:

$$\text{At } t = 0.9s, P_{safe} = P(x_2 = Safe \mid \mathcal{Z}_2) = 0.9993. \quad (15)$$

Case C. Time $t = 1.3s$. Note that, at this time, there is no observation from either modality. We can still compute P_{safe} using (4), as follows:

$$P(x_3 \mid \mathcal{Z}_2) = \sum_{x_2 \in \mathcal{S}} P(x_3 \mid x_2)P(x_2 \mid \mathcal{Z}_2).$$

Using (15) and the HMM transition probabilities, we may expand to get:

$$\begin{aligned} P(x_3 = Safe \mid \mathcal{Z}_2) &= P(x_3 = Safe \mid x_2 = Safe) \\ &\quad P(x_2 = Safe \mid \mathcal{Z}_2) + \\ &\quad P(x_3 = Safe \mid x_2 = Attacked)P(x_2 = Attacked \mid \mathcal{Z}_2) \\ &= e^{k\Delta t} \times P(x_2 = Safe \mid \mathcal{Z}_2) + \\ &\quad 0 \times P(x_2 = Attacked \mid \mathcal{Z}_2) \\ &= 0.9908 \times 0.9993 = 0.9901 = P_{safe}, \end{aligned}$$

where $\Delta t = 1.3 - 0.9 = 0.4$.

In summary, this simple numerical example demonstrates how our Holistic Fusion method:

1. Combines face and fingerprint observations in the order in which they occurred.
2. Accounts for the reliability of different modalities by using the intra and interclass pdfs. This satisfies Criterion 1.
3. Discounts older observations by using transition probabilities that decay as a function of the elapsed time between observations. This satisfies Criterion 2.
4. Computes P_{safe} at any time, whether or not there are biometric observations. This satisfies Criterion 3.

ACKNOWLEDGMENTS

This work is funded by the National University of Singapore, Project #R-252-000-146-112. The authors would like to thank Dr. Roland Yap and Mr. Jerel Yam for their assistance in the Windows XP implementation.

REFERENCES

[1] L. Joncheray, "A Simple Active Attack Against TCP," *Proc. Fifth USENIX Security Symp.*, pp. 7-19, 1995.
[2] A. Ross and A.K. Jain, "Information Fusion in Biometrics," *Pattern Recognition Letters*, vol. 24, no. 13, pp. 2115-2125, 2003.
[3] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas, "On Combining Classifiers," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226-239, Mar. 1998.
[4] L. Kuncheva, "A Theoretical Study on Six Classifier Fusion Strategies," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 281-286, Feb. 2002.
[5] R.P.W. Duin and D.M.J. Tax, "Experiments with Classifier Combining Rules," *Proc. First Workshop Multiple Classifier Systems*, pp. 16-29, 2000.
[6] A. Altinok and M. Turk, "Temporal Integration for Continuous Multimodal Biometrics," *Proc. Workshop Multimodal User Authentication*, pp. 131-137, 2003.
[7] L.R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257-286, 1989.
[8] A. Klosterman and G. Ganger, "Secure Continuous Biometric-Enhanced Authentication," Technical Report CMU-CS-00-134, Carnegie Mellon Univ., May 2000.
[9] A.G. Morgan, "The Linux-PAM System Administrators' Guide," documentation distributed with Linux-PAM, http://www.kernel.org/pub/linux/libs/pam/pre/library/, 2006.
[10] C. Carrillo, "Continuous Biometric Authentication for Authorized Aircraft Personnel: A Proposed Design," master's thesis, Naval Postgraduate School, 2003.
[11] R. Janakiraman, S. Kumar, S. Zhang, and T. Sim, "Using Continuous Face Verification to Improve Desktop Security," *Proc. IEEE Workshop Applications of Computer Vision*, pp. 501-507, 2005.
[12] "The Linux Kernel Archives," http://www.kernel.org/, 2003.
[13] S. Kumar, T. Sim, R. Janakiraman, and S. Zhang, "Using Continous Biometrics Verification to Protect Interactive Login Sessions," *Proc. 21st Ann. Computer Security Applications Conf.*, pp. 441-450, 2005.

[14] S. Zhang, R. Janakiraman, T. Sim, and S. Kumar, "Continuous Verification Using Multimodal Biometrics," *Proc. Second Int'l Conf. Biometrics,* pp. 562-570, 2006.
[15] P. Viola and M. Jones, "Robust Real-Time Object Detection," *Int'l J. Computer Vision,* 2002, citeseer.ist.psu.edu/viola01robust.html.
[16] T. Sim, R. Sukthankar, M. Mullin, and S. Baluja, "Memory-Based Face Recognition for Visitor Identification," *Proc. IEEE Int'l Conf. Automatic Face and Gesture Recognition,* pp. 214-220, 2000.
[17] R. Duda, P. Hart, and D. Stork, *Pattern Classification,* second ed. John Wiley and Sons, 2000.
[18] N. Crook, "The kdm Handbook," http://docs.kde.org/en/3.1/kdebase/kdm/, 2005.

Terence Sim received the PhD degree in electrical and computer engineering from Carnegie Mellon University in 2002, the MS degree in computer science from Stanford University in 1991 and the BS degree in computer science and engineering from the Massachusetts Institute of Technology in 1990. He is an assistant professor at the School of Computing, National University of Singapore. His research interests are in biometrics, face recognition, computer vision, digital photography, and music processing. He also chairs the workgroup on cross-jurisdictional and societal aspects in the Biometrics Technical Committee, Singapore. He is a member of the IEEE.

Sheng Zhang received the BS degree from Zhejiang University, China, in 1998 and the MS degree from the Institute of Automation, Chinese Academy of Sciences in 2001. He is now a PhD candidate at the School of Computing, National University of Singapore. His research interests include: face recognition, face space, statistical pattern recognition, and machine learning. He is a student member of the IEEE and the IEEE Computer Society.

Rajkumar Janakiraman received the BS degree in information technology from the University of Madras, India, in 2003 and the MS degree in computing from the National University of Singapore in 2006. He currently works as a research assistant under the supervision of Dr. Terence Sim. His research interests include: biometrics, computer vision, and statistical pattern recognition.

Sandeep Kumar received the BS degree in electrical engineering from the Indian Institute of Technology, New Delhi, and the PhD degree in computer science from Purdue University. He is currently a staff researcher at General Motors Labs in Bangalore, India. Prior to joining GM, he was an assistant professor at the School of Computing, National University of Singapore, where this work was completed.