Continuous User Identity Verification for Trusted Operators in Control Rooms

Enrico Schiavone (✉), Andrea Ceccarelli, Andrea Bondavalli

Department of Mathematics and Informatics, University of Florence, 50134 Florence, Italy

{enrico.schiavone,andrea.ceccarelli,bondavalli}@unifi.it

Abstract. Human operators in control rooms are often responsible of issuing critical commands, and in charge of managing sensitive data. Insiders must be prevented to operate on the system: they may benefit of their position in the control room to fool colleagues, and gain access to machines or accounts. This paper proposes an authentication system for deterring and detecting malicious access to the workstations of control rooms. Specifically tailored for the operators in the control room of the crisis management system Secure!, the solution aims to guarantee authentication and non-repudiation of operators, reducing the risk that unauthorized personnel (including intruders) misuses a workstation. A continuous multi-biometric authentication mechanism is developed and applied in which biometric data is acquired transparently from the operator and verified continuously through time. This paper presents the authentication system design and prototype, its execution and experimental results.

Keywords: Biometrics; Verification; Trust; Security; Control Rooms.

1 Introduction

Secure user authentication is fundamental for several ICT (Information and Communication Technology) systems. User authentication systems are traditionally based on pairs of username and password and verify the identity of the user only at login phase. No checks are performed during working sessions, which are terminated by an explicit logout or expire after an idle activity period of the user. While this is often sufficient, it may not result enough against *insider attackers* [14, 18] in control rooms, where operators are using their workstation to access potentially sensitive data and to issue critical commands for the entire working session; the operators are directly responsible for such commands and for the data accessed, modified and deleted.

In this paper we consider the behavior and actions of the human operators working in the control room of the Secure! [1] *crisis management system*. Such operators are in charge of analysing and interpreting situations that describe the current status of an emergence. Using the information available, the operator from his workstation (mainly via text messages, using a keyboard) is able to command intervention teams

on field, and to dispatch instructions to civilians in the target area. More details regarding the objectives of the Secure! project and its resulting system are available in [1].

It is required to protect the control rooms and its workstations from unauthorized people (*intruders*) and *insiders* that may want to acquire privacy-sensitive data, disrupt the crisis management operations, disseminate false information, or simply commit errors, which will be ascribed to the operator in charge of the workstation.

Consequently, in order to protect the workstations, we need to guarantee (i) *authenticity* of the commands/functions executed, meaning that commands that are transmitted and expected from an operator, are actually generated from him, and (ii) *non-repudiation* of the commands/functions executed, meaning that the worker which sends the commands from a workspace is known.

To timely detect misuses of computer resources and prevent that an unauthorized user maliciously replaces an authorized one, solutions based on *biometric continuous authentication* [2] are proposed in literature, turning user verification into a continuous process rather than a one time occurrence [3]. Additionally, to improve its security, biometrics authentication can rely on multiple biometrics traits thus, being *multi-modal*. Finally, biometric data can be acquired *transparently* i.e., without explicitly notifying the user or requiring his/her interaction, aiming to improve service usability [5].

In this paper, we investigate a *continuous multi-modal biometric authentication* protocol for *transparent* verification of the operator identity in the Secure! control room, concretely presenting how to implement the approach on a real life case study. Starting from an analysis of solutions available in the state of the art, we tailored a solution that integrates face, fingerprint and keystroke recognitions. Face data is acquired using a camera, fingerprint data is acquired via a fingerprint sensor integrated in the mouse, and keystroke data is instead acquired via the keyboard.

The protocol removes the necessity of explicit interactions to prove the operator identity, and thanks to the multi-modality, it allows acquiring biometric data also when different operators are using the same workstation. For example, if more operators are in front of the camera thus in some cases compromising the face recognition, the legitimate operator can still use the fingerprint reader. Finally, authenticity and non-repudiation are guaranteed by the continuous authentication, which is intended to assure that the operator is within range of the workstation during its use.

The rest of the paper describes in Sect. 2 some of the available solution from the state of the art and the advancements of our work, in Sect. 3 the design of our tailored solution and in Sect. 4 the realization of our prototype. Results on its execution are reported in Sect. 5. Finally, conclusions are in Sect. 6.

2 Multi-modal Biometric Continuous Authentication

Biometrics refers to a measurement of physiological and/or behavioral characteristics of the human body; a biometric recognition system provides an automated method for confirming (*verification*) or determining (*identification*) the identity of an individual

based on his characteristics [7]. Identity verification, which is the target of our work, consists of a one-to-one matching and occurs when an individual claims his identity. The system needs to compare the newly acquired biometric data and the previously enrolled digital representation of an individual's biometric characteristics, usually called templates. It is well-known that using multiple biometric characteristics combined with an appropriate rule, that is, providing a multi-modal biometric authentication, can yield a higher performance than using only one trait [15, 16].

Surveying the state of the art, a large number of studies can be identified regarding biometric continuous authentication. We review approaches on biometric authentication systems based on *multi-modality* and *continuous verification* where a *transparent* acquisition of biometric data is researched. Each of them could be used for our purpose but for each one we can find at least one reason that suggests the introduction of a new approach, ad-hoc for our requirements.

The work in [2] describes a multi-modal biometric passive continuous authentication system, combining face and fingerprint recognition to verify the physical presence of a user. The authors state that it introduces a significant overhead (between 26% and 42%) and the user's task are delayed; the reason is probably the bottleneck generated by a too frequent acquisition of the face (two times for each second) and fingerprint images (once per second).

The work in [3] proposes a multi-modal continuous biometric authentication system integrating information temporally as well as across multiple biometric modalities. The main idea of the method is based on the assumption that as time passes, the authentication system is less and less certain about the authentication score value. Experiments show that temporal information improves authentication accuracy. However, the acquisition of 15 images in less than a second suggests that the impact of this solution in terms of computational resource usage would be relevant, and seems legitimate to expect it would weigh down the system.

In [4] the objective is to investigate the opportunity of using a multimodal biometric system as input of a fuzzy controller for preventing user substitution after the initial authentication process. The chosen modalities are face and fingerprint. The role of the fuzzy controller is to request the fingerprint data only if the face recognition matching produces a trust level that is below a threshold. Nevertheless, we need a transparent acquisition of the traits and the explicit request of the fingerprint does not meet this requirement.

Finally, the work in [12, 13] proposes a continuous multi-modal sequential biometric authentication solution, where trust in the user is computed after each successful user identity verification and it is decreased as time passes without successful verifications. An authentication server is in charge of receiving biometric data, performs verifications and computes trust in the user. We adopt the trust formulation from [12, 13] because of its simplicity and easiness to adapt to different systems and sensors. Relevant tailoring w.r.t. [12, 13] was needed due to the different requirements of our system, in fact we are considering a control room with workstations and a defined set of sensors, selected on the basis of the usual actions of trained operators, rather than potentially any environment, kind of device, and user. For example, the operator is expected to have his hand on the mouse for most of the time, and this leads to introducing a fingerprint reader in the mouse. In addition, our system

automatically grants access to all critical functions after login, while [12, 13] protects communication towards each specific service individually.

3 Our Approach to Continuous Authentication

The overall architecture of the biometric system is composed of the operators' workstations and the connected sensors required to acquire the biometric data. Biometric data are transmitted to an authentication system, which includes a database with the biometric templates of the operators.

3.1 The protocol

In our protocol, the different biometric data are acquired continuously by the workstation, and the identity of the operator is verified; an estimation of the trust in the operator is then computed. Such trust is described as a value ranging through time in the interval [0; 1], and that decreases through time, at different rating speeds, depending on the action and behaviour of the operator (w.r.t. the available sensors). The trust value increases only when fresh biometric data is acquired and successfully verified. When such trust value is lower than a given threshold, the permissions of the operator are reduced thus limiting the possible actions that they can execute on the Secure! system until a new login is performed.

Considering the operator of the Secure! system and comparing a set of well-known biometric traits [6], we selected the following three traits for multimodal biometric authentication. First, fingerprints are acquired using a sensor integrated in the mouse [8] that will be described in Section 4, thus allowing fingerprint recognition. Second, facial images are acquired using a camera (a webcam) allowing face recognition. Third, keystroke data are acquired with keyboard allowing keystroke recognition. The objective is to combine the transparent and sequential acquisition of the above data to continuously assure trust in the operator.

The three above biometric traits have different levels of performance and measurability [6, 7] and complement each other. High measurability of facial images will help covering temporal gaps that could exist between two fingerprint acquisitions. Keystroke supports the other two traits despite its low performance. Especially keystroke can results useful when fingerprint acquisitions are missing e.g., because the operator is not touching the mouse: in fact, when the operators are typing on the keyboard, they are most likely unable to place their finger on the fingerprint reader.

Thus, we introduce a mechanism where (i) keystroke recognition on the text typed is executed in order to recognize the operator, and (ii) the usage of the keyboard is considered a justification for the absence of fingerprint acquisition. All these considerations will influence the computation of the trust in the operator.

In the rest of the paper, we present the implementation of the authentication system, where we choose three exemplary recognition algorithms, but we imagine that our method can work efficiently also with different face, fingerprint and keystroke recognition algorithms, and they can be changed if necessary.

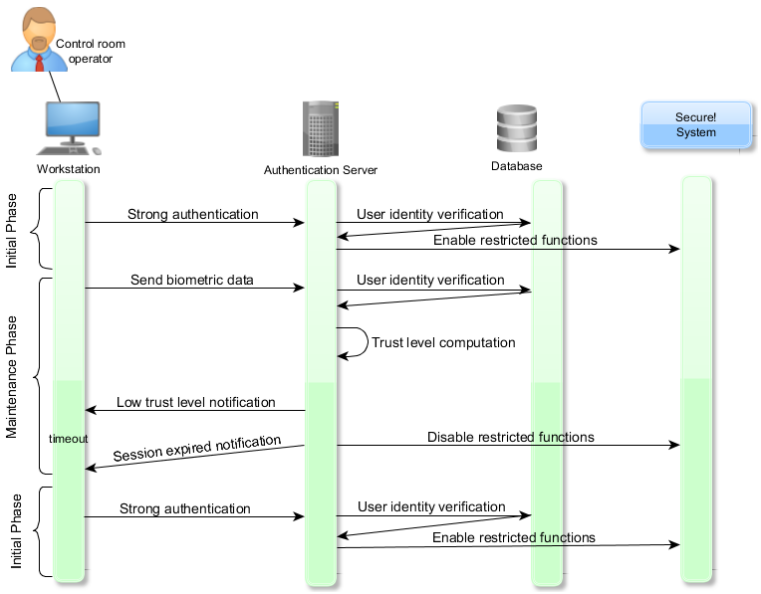

Fig. 1. Sequence diagram of the protocol

The proposed continuous authentication protocol is shown in the sequence diagram of Fig. 1. It is based on three biometric subsystems, one for each trait, where each subsystem is composed of hw/sw elements necessary for the acquisition of biometric traits and for the verification process, including sensors and recognition algorithms.

The protocol is divided in two phases: the initial phase and the maintenance phase.

Initial phase. It is composed of the following steps:

- The user logs in to start a session that will also imply the possibility of using functionalities that need authentication. A strong authentication is here needed for login, with a one-time password (a password that is valid for only one login session or transaction) or by a successful biometric verification executed with all the three subsystems in a short time interval.
- Data are acquired by the workstation and transmitted to the authentication server.
- The authentication server uses the operator's templates contained in a biometric database and verifies his identity. In case of successful verification, the authentication server communicates to the Secure! system to establish a session and to allow all restricted functions expected for the operator's role. In addition, the authentication server computes and updates a trust level that decreases as time passes (trust computation is described in the next subsection); the session will expire when such level becomes lower than a threshold. The maintenance phase is started to compute a new trust level.

Maintenance phase. The biometric continuous authentication protocol works as follows:

- The authentication server waits for fresh biometric data, from any of the three subsystems. No active participation of the operator is necessary, which only needs to use the mouse, the keyboard or to be positioned in front of the camera.
- When new biometric data is available for the different biometric subsystems, the authentication server verifies the identity claimed by the operator and, depending on the matching results of each subsystem, updates the trust level.
- When the trust level is close to the threshold, the authentication server sends a notification to the operator, to signal that in case biometric data is not transmitted, the session will expire soon.
- If the trust level is below the threshold, the authentication server communicates to the Secure! system to disable the restricted functions and notifies the operator that his session is expired. In this case, the operator can access only a set of non-restricted functions, i.e., functions with no or reduced criticality in terms of privacy and confidentiality.

The disabled functions will be available again only when the operator performs a strong authentication, restarting from the initial phase.

3.2 Internals: the trust level computation

We now introduce some concepts useful to describe the algorithm for trust level computation, without the intent of being generic but focusing the discussion on the specificities of our system and the biometric traits used. A generic approach can be derived from the discussion below whenever needed.

Given the three unimodal biometric subsystems S_1=fingerprint recognition, S_2=face recognition, S_3=keystroke recognition such that each one is able to decide independently if the user is genuine or not, we define $m(S_1)$, $m(S_2)$, $m(S_3)$ as the *trust in the subsystem* respectively S_1, S_2, S_3. The $m(S_1)$, $m(S_2)$, $m(S_3)$ are static values that lie in the interval [0,1]; the more trust we place in a subsystem, the higher its value.

We also define the trust level $trust(u, t)$ that represents the trust put in the user u at time t by the authentication server. In other words, it corresponds to the probability that the operator u is legitimate considering his behavior using the workstation. It takes into account the time from last acquisition of biometric data, and the combination of the individual decisions of the three subsystems S_1, S_2, S_3. Being considered as a probability, it is a value that lies the interval [0, 1].

Finally, we define a lower bound $trust_{min}$ corresponding to the minimum threshold of the trust level requested by the authentication system. If $trust(u, t) < trust_{min}$, the authentication server disables the restricted functions. If instead $trust(u, t) \geq trust_{min}$ the user is maintained authenticated and the access to all restricted functions is granted. To ease the readability of the notation, in the following the operator u is often omitted.

Table 1. Example of the trust level computation

Pair of biometric subsystems	Trust level computed as in equation (1)
S_{k1}, S_{k2}	0.9 + (0.1· 0.8) =0.98
S_{k1}, S_{k3}	0.9 + (0.1· 0.7) =0.97
S_{k2}, S_{k3}	0.8 + (0.1· 0.7) =0.87

The algorithm for the computation of the trust level is executed iteratively on the authentication server as explained in what follows. In the *initial phase*, at time t_0, the operator performs a strong authentication: if successful, the trust is set to $trust(t_0) = 1$.

The maintenance phase is here started. Each subsystem continuously tries to acquire, on the workstation, the biometric data of the operator and transmits them to the authentication server. The authentication server verifies the operator identity using all biometric data provided in a specific time interval; let us consider the time interval $[t_{i-1}; t_i]$ and consequently we reason for the status of the system at time instant t_i. We have the following options at t_i. In the first case, all the three biometric subsystems led to successful verification: the trust level is set to $trust(t_i) = 1$. In the second case, two of the three biometric subsystems led to successful verification. This means that one biometric subsystem could not acquire data or decided that the operator is not legitimate. The trust level is computed following (1):

$$trust(t_i) = m(S_{k1}) + \big(r \cdot m(S_{k2})\big). \quad (1)$$

Where:

- S_{k1} and S_{k2} are the two subsystems, which correctly verified the identity of the operator, and S_{k2} is the one with the lower performance;
- r is a parameter that allows to weight $m(S_{k2})$ in order to have $trust(t_0)$ between 0 and 1.

For example, setting $r = 0.1$, $S_{k1} = 0.9$, $S_{k2} = 0.8$, $S_{k3} = 0.7$, we have the combinations reported in Table 1. The selection of these values can be conducted comparing the biometric traits, and from an analysis of their performance, i.e., in terms of number of false positives and false negatives produced by each. We found that these values can represent properly the accuracy of each subsystem, but other different values can be easily adopted, if necessary, following a similar approach. Until at least two subsystems do not decide that the user is legitimate in the same time interval, thus updating $trust(t_i)$ following Table 1, the trust level decreases and the session will expire when it is smaller than $trust_{min}$. If at t_i at most one biometric verification is successful (e.g., for two biometric subsystems no biometric data is acquired, or verifications failed), the trust level $trust(t_i)$ is computed using the following.

Supposing we have $trust(t_{i-1})$ that is, the trust level computed at the previous iteration of the algorithm, we want to compute the new $trust(t_i)$, which will be smaller than $trust(t_{i-1})$. Therefore, at time t_i, the trust level is given by (2), [13]:

$$trust(t_i) = \frac{(-\arctan\big((\Delta t_i - s) \cdot k\big) + \frac{\pi}{2}) \cdot \mathrm{trust}(t_{i-1})}{-\arctan(-s \cdot k) + \frac{\pi}{2}}. \quad (2)$$

Where $\Delta t_i = t_i - t_{i-1}$, and parameters k and s are introduced to tune the decreasing function: k affects the inclination towards the falling inflection point and s allows anticipating or delaying the decay.

The selection of k in particular affects the speed of the decrease of the trust level. We adopt three different values of k in order to provide three different kinds of decrease, described below. A *fast* decrease (k=0.01) is set when no verifications are successful or no biometric data is transmitted. In this case, the trust will rapidly decrease towards $trust_{min}$. An *average* decrease (k=0.001) is set if only one verification is successful, for any biometric subsystem. Finally, a *slow* decrease (k=0.0008) is set if face verification is successful and the usage of keyboard is detected, although data is not sufficient to perform keystroke recognition or keystroke recognition fails. This situation means that the Secure! operator is actually busy using the keyboard and he cannot send any fingerprint data, and that the amount of keys pressed is too low or too sparse to permit keystroke recognition. Thus, a small penalization is assigned to the trust in the operator, smoothly decreasing the trust level.

4 The prototype

4.1 Hardware prototype

Our prototype is composed of two PCs. On the client side, the workstation that we are using in our Secure! prototype is a Fujitsu Lifebook A-530 with an Intel® Pentium® P6200, 4GB RAM, and running Windows 7, equipped with the following biometric sensors. For fingerprint acquisition, our choice is the SecuGen OptiMouse Plus mouse [8], which incorporates an optical fingerprint scanner at the place where a user would normally place their thumb. Such fingerprint scanner does not require active participation by the user, and therefore does not require that the operators periodically perform biometric-related tasks that are not part of their normal activities. For acquisition of the images for face recognition, we use the built-in camera of the workstation that can continuously capture images without the active cooperation of the user. For the acquisition of keyboard data, we collected them using the standard PS/2 keyboard integrated in the Fujitsu Lifebook. As authentication server, we are using an HP Pavilion Desktop PC500-420nl with processor i5-4460S and 8 GB RAM which is in the same local network of the workstation.

4.2 Software design

In this section, we describe our software implementation. All software we developed is implemented in Java. Client-server communication is based on RESTful web services, and developed using the Jersey framework, following the design specification of the Secure! project.

The workstation software is started by a *Client* object, which activates the methods of (i) *InvisibleFaceTracker*, (ii) *KeyListener* and (iii) *FingerPrintDetection*. Each class contains respectively: (i) an algorithm that, exploiting the camera, saves an image if (at least one) face is detected, (ii) a procedure that, exploiting the SecuGen OptiMouse Plus, cyclically detects and saves a fingerprint which, when available, is

transmitted to Client, (iii) an algorithm that detects the pressing of the keys. At fixed time intervals, a text file is saved containing, for each row, the press and release times for each pressed key. Such file is delivered to the authentication server. The *Client* is in charge of invoking the *UploadFileService* REST client to transmit via HTTP post the saved image, fingerprint and text file to the authentication server together with the client ID.

On the authentication server, the *TrustCalcService* guides the biometric verification, the calculation of the trust level, and the communication of session expiration to the Secure! system and to the workstation. The *TrustCalcService* class contains the RESTful web service that receives the transmitted biometric traits from the client, and a REST client to communicate the session expiration to the workstation. Getting the information from the client, the web service decides which methods of the *RecognitionHandler* class should be called to start the verification process. Each subsystem produces a decision about the legitimate of the user, as explained in Sect. 3. At fixed time intervals, the trust level is raised according to Table 1. Example of the trust level computation or, if less than two traits are correctly verified, selecting the trust decaying function with the appropriate k value as in Eq. (2).

However, for a final product realization, the implementation should integrate recognition algorithms with very high performance. Moreover, depending on the specific characteristics of the algorithms, the parameters and the time intervals have to be tuned properly.

4.3 Enabling technologies

The SecuGen's FDx Software Developer Kit [8] provides low-level APIs for device initialization, fingerprint capture and matching functions. In the enrollment phase, templates are computed and stored in raw format.

We customized the face recognition software available in [10]; this is able to (i) analyze the frames captured via a camera, (ii) locate a face in the frames, and then (iii) verify user's identity. When a face is present in front of the camera, the algorithm [10] detects its presence; in most of the cases, this happens within approximately 40 ms. Otherwise, if a face is not present, the algorithm takes up to 200 ms to ultimately notify that no face is present. The implementation available in [10] requested the installation and configuration of OpenCV [9], an open source library that includes several hundreds of computer vision algorithms, and of JavaCV, the related Java interface. Our customization of the software was necessary in order to i) structure the implementation available in [10] in two client and a server sides, where the first is in charge of capturing images and deciding if a face is present, and the second performs verification, and ii) make the acquisition of the biometric data transparent and automatic, removing the graphical interface and interactions of the user with the software. An enrollment phase is obviously required, in which the operator ID is associated to a set of face templates. The verification phase compares the selected face to the enrolled templates to produce a matching result.

Keystroke data acquisition relies on the library JNativeHook that provides keyboard (and mouse) listeners for Java. In particular, this library allows detecting keys press and release events and captures, in correspondence to those events, the time

instant of the events. JNativeHook also permits to detect the keyboard usage (and the keys pressed), both if the user is typing in a specific text area or not: the cursor position is not relevant. This is consistent with our needs as we can capture keystroke data without being invasive for the activity of the control room operator.

Relying on such library, we realized the keystroke recognition in the *KeyAnalyzer* class, implementing the algorithm described in [11]. Such algorithm continuously collects the keystroke dynamics (the typed key and related pressing and release time) and applies a penalty/reward function on the dataset to measure the confidence that the user has not changed in the selected time interval. An enrollment session is required where the operator types several sentences, to create a biometric template based on the timing information for each typed key and key combination [11].

In our implementation of [11], when the keyboard is used with continuity i.e., there is evidence that someone is currently typing, we collect keystroke dynamics for a defined time interval and then we transmit all values to the authentication server. The selection of the time interval is critical because if the number of values collected is too low, verification will most likely fail: a short time interval would probably result ineffective for keystroke authentication following [11]. Moreover, a long time interval would imply a long wait before transmitting the values, thus risking that the session expires meanwhile. We evaluated that listening for up to 10 seconds of continuous typing was deemed sufficient to allow successful verification (we also remember that keystroke is the weakest of our biometric traits, and it is easily prone to false positives).

4.4 Availability, security, privacy and performance

The Secure! system implements solutions for the overall availability of the system, the security and the privacy of the information managed, stored or exchanged, following a threat and risk analysis that was performed at the beginning of the project [14]. The mitigations identified for risks and threats, including time-related ones, also consider the workstations, the authentication server, and the related communication channels. Although such analysis and the Secure! architecture are not within the scope of this paper, we present considerations on availability, security, privacy and performance that we believe relevant.

Regarding availability, our authentication mechanism clearly requires that the authentication server and all communications channels in Fig. 1 are up and running. In case of unavailability of any of the above, the system administrator of the Secure! system is able to temporarily disable the continuous authentication, thus switching to a traditional password-based authentication approach. However, such alternative should be exploited only when needed and matched to immediate intervention for maintenance (strategies for rapid maintenance intervention are foreseen for the whole Secure! system).

Protection of the biometric data exchanged and stored, together with protection of the communication between the entities of Fig. 1 is mandatory. Briefly, the system in Fig. 1 represents a closed system, where all interacting entities are known, all communications are cabled, and no external machines are accepted. Communications are ciphered, and access to the entities is protected; solutions for the protection of data

and communications are defined and applied to the whole Secure! system, as it manages several other sensitive and secure data, in addition to the biometric ones.

Privacy of data is fundamental in Secure! both for data related to crisis management and for data related to the continuous authentication, which describes the behaviour of the worker in the operating room. Although data management is part of the Secure! architecture, it is worthy to discuss the authentication data. Such data is stored for a limited time (few days), and then removed, thus the system maintains only the recent history on the behaviour of the user. Access to such history is regulated by the procedures of the operating room and it is allowed only to investigate on suspected security breaches.

Regarding performance, the continuous authentication software executing on the workstation may potentially slow down the Secure! application used by the operator. We measured the overhead introduced on our prototype, resulting in approximately an increase of usage of 6% for CPU and 2% for RAM. Such overhead is limited and shall not affect the execution of the Secure! application. The authentication server is instead in charge of managing the biometric database and verifying the identities of the team of operators, thus it is subject to a relevant computational load especially for large teams. However, the authentication server is one of the (powerful) nodes of the Secure! framework, which has been built with scalability [17] in mind so that its nodes can be easily adapted to sustain high computational loads.

4.5 Usability

We comment on the impact of our authentication solution on the daily activities of the operator. The authentication solution and the values of its parameters were selected to achieve a compromise between security (a malicious operator is disconnected in at worst approximately 40 s, see Fig. 3) and usability (the worker maintains the session active, mostly thanks to the fingerprint reader in the mouse, which is used most of the time, and the face recognition). The experiments in Sect. 5, which adopt the configurations selected for Secure!, confirm that a worker is able to maintain the authentication until he voluntarily leaves the workstation. We are aware that usability studies in daily working sessions with multiple Secure! operators are required; this is the main objective of the ongoing experimentation of our solution.

5 Explanatory Results

We report on three typical scenarios where 10 runs have been performed using the prototype and the data for one case are plotted. The configurations selected for Secure! are the same as Sect. 3; trust threshold is set to $trust_{\min} = 0.5$. The objectives of the three scenario analysis are respectively i) verify that the operator working at a workstation is able to maintain authentication for a whole working the workstation is left unattended by the legitimate user, the trust will decay below the thresholds, iii) verify the requirements on the environment, especially on illumination.

In the *first scenario,* an operator is in front of the client, working in an environment with good illumination. A working session of 50 minutes is performed: the operator

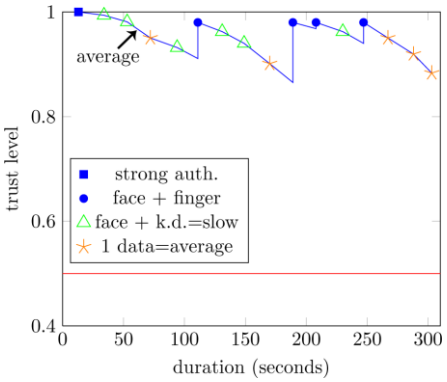

Fig. 2. Successful verifications: trust level is always above the threshold

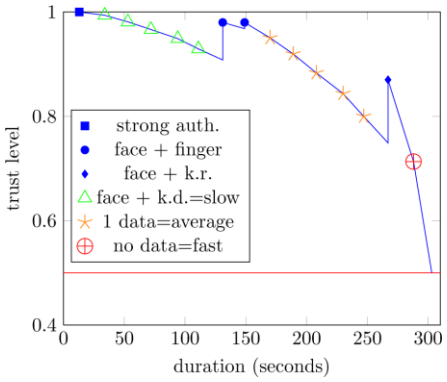

Fig. 3. Operator leaving the workstation unattended after a while.

does not leave the workstation, and alternates the usage of keyboard and mouse to perform his work. An extract showing the initial part of one run is shown in Fig. 2 to clarify the behavior of the protocol. At the beginning of the run (at second 13), a strong authentication is successfully performed providing in a time slot the three biometric traits (the time slot is set to 10 seconds): this means that the three biometric data should be provided and successfully verified in such interval. Note that additional delays due to transmission and processing time should always be taken into account. After the initial authentication, at time intervals of around 20 seconds, the authentication server verifies the biometric data. The first and second time intervals resulted in successfully performed verification for face and detection of the usage of keyboard (keyboard detection, or k.d., in Figs 2 and 3), leading to a slow decrease of trust level with k=0.0008. In the third time interval, only face is successfully verified, with no usage of keyboard detected. This leads to an average decrease of trust level, with k set to 0,001. For clarity, this interval is also identified in Fig. 2 with the arrow with label "average". In the fifth time interval, at second 111 face and fingerprint subsystems session¸ ii) verify that if successfully verify the operator identity, resulting in raising the trust level to 0,98 according to Table 1. Note that in Fig. 2 every time the operator is using the keyboard the trust level drops, in fact the usage of the keyboard was detected (keystroke detection), but keystroke recognition failed or was not performed due to insufficient data. Then the algorithm follows in a similar manner alternating different successful verifications in the time intervals for the whole duration of the experiment, with a predominance of face recognitions (the cases in which only one biometric trait is verified always refer to face). On the whole set of 10 runs, 1 facial image per second was acquired, and if face was detected, the algorithm processed the image. This resulted in an average of 5 missed face recognition per run (obviously, this is strictly dependent on the face recognition algorithm). Fingerprint lead only to an average of 2 mismatches out of approximately 200 checks per run, while keystroke lead to an average of 15 mismatches out of 24 attempts per run.

The *second scenario* aims to show the possible behavior of the prototype when the operator leaves the workstation unattended, and consequently it is expected that the session will terminate due to failed biometric recognition or no acquisition of biometric data. In this experiment, we observe that ultimately the trust level decays

below the threshold, resulting in session expiration. We show a sample run of the experiment in Fig. 3, as the others behave similarly. A strong authentication is initially performed with the same outcome as in the previous experiment. Then the trust level slowly decreases, given the recognition of face and the usage of keyboard, until second 131 in which the two biometric traits face and fingerprint are correctly acquired and verified. These are newly verified at second 149. Then the decay of the trust level is set to average because only the face is verified, until second 267 when the trust level is raised thanks to identity verification via face and keystroke recognition (e.g., the operator after reading some text was able to write an answer). The resulting trust is set to 0.87 following Table 1. At this point, no more biometric verification are successful, resulting in a fast decrease of the trust that leads to reaching the timeout in around 36 seconds. The threshold is ultimately reached at second 303. Such timeout is considered adequate for the purpose of Secure!, as in general this time is too short to have the operator leave the workstation unattended and execute commands or acquire data.

Finally, the last scenario is conducted in a room where the only source of lighting is a window placed behind the operator. The runs performed led to an unexpected termination of the session, because the facial recognition failed repeatedly leading to a decay of the trust thresholds. The implication is that it is required to set the control room environment appropriately in order to have our solution work properly.

6 Conclusions

This paper presented our realization of a continuous multi-modal biometric authentication system for the operator active in the Secure! control room. The protocol is able to transparently acquire face, fingerprint and keystroke traits to continuously verify the identity of the operator without his explicit involvement. The paper described the solution, its prototype realization and execution. Results show that, despite obvious limitations due to the necessity of continuously provide biometric data, when appropriately tailored for a working environment our solution allows maintaining the worker authenticated and improving system security.

As future work we are building and executing several test sessions with a larger number of different operators using Secure!, to investigate the tradeoff between security and usability for the different settings of the authentication system.

Acknowledgments This work has been partially supported by the POR-CREO 2007-2013 Secure! project funded by the Tuscany Region, by the European FP7-IRSES project DEVASSES, and by the TENACE PRIN Project (n. 20103P34XC) funded by the Italian Ministry of Education, University and Research.

References

1. Secure! Project, http://secure.eng.it
2. Kumar, S., Sim, T., Janakiraman, R., Zhang, S: Using continuous biometric verification to protect interactive login sessions. In: 21st Annual Computer Security Applications Conference (ACSAC), pp. 441-450 (2005)
3. Altinok, A., Turk, M.: Temporal integration for continuous multimodal biometrics. In: Proceedings of the Workshop on Multimodal User Authentication (2003)
4. Azzini, A., Marrara, S., Sassi, R., Scotti, F.: A fuzzy approach to multimodal biometric continuous authentication. In: Fuzzy Optimization and Decision Making, 7(3), pp. 243-256 (2008)
5. Crawford, H., Renaud, K., Storer, T.: A framework for continuous, transparent mobile device authentication. In: Computers & Security, 39, pp. 127-136 (2013)
6. Jain, A. K., Ross, A., Prabhakar, S.: An introduction to biometric recognition. In: IEEE Transactions on Circuits and Systems for Video Technology, 14(1), pp. 4-20 (2004)
7. Tripathi, K. P.: A comparative study of biometric technologies with reference to human interface. In: International Journal of Computer Applications, 14(5), pp. 10-15 (2011)
8. SecuGen OptiMouse Plus, http://www.secugen.com/products/po.htm.
9. The OpenCV Reference Manual, Release 2.4.9.0, (2014)
10. Davison, A.: Killer Game Programming in Java. O'Reilly Media Inc. (2005)
11. Bours, P., Barghouthi, H.: Continuous Authentication using Biometric Keystroke Dynamics. In: The Norwegian Information Security Conference (NISK), (2009).
12. Ceccarelli, A., Bondavalli, A., Brancati, F., La Mattina, E.: Improving Security of Internet Services through Continuous and Transparent User Identity Verification. In: IEEE 31st Symposium on Reliable Distributed Systems, (SRDS), pp. 201-206 (2012)
13. Ceccarelli, A., Montecchi, L., Brancati, F., Lollini, P., Marguglio, A., Bondavalli, A.: Continuous and Transparent User Identity Verification for Secure Internet Services. In: IEEE Transactions on Dependable and Secure Computing, 12(3), pp. 270-283 (2015)
14. Nostro, N., Ceccarelli, A., Bondavalli, A., Brancati, F.: Insider threat assessment: A model-based methodology. In: Operating Systems Review (ACM), 48 (2), pp. 3-12 (2014)
15. Ross, A., Jain, A. K.: Information fusion in biometrics. In: Pattern Recognition Letters, 24(13), pp. 2115–2125 (2003)
16. Hong, L., Jain, A. K., Pankanti, S.: Can Multibiometrics Improve Performance?. In: Proceedings AutoID, (99), pp. 59-64, (1999)
17. Montecchi, L., Nostro, N., Ceccarelli, A., Vella, G., Caruso, A., Bondavalli, A.: Model-based Evaluation of Scalability and Security Tradeoffs: a Case Study on a Multi-Service Platform. In: Electronic Notes in Theoretical Computer Science, 310, pp. 113-133 (2015)
18. Nostro, N., Ceccarelli, A., Bondavalli, A., Brancati, F.: A methodology and supporting techniques for the quantitative assessment of insider threats. In: Proceedings of the 2nd International Workshop on Dependability Issues in Cloud Computing (ACM), (3), pp. 1-6 (2013)