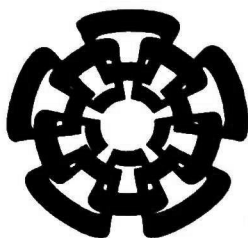


XX(131348 1)



CINVESTAV

Centro de Investigación y de Estudios Avanzados del I.P.N.
Unidad Guadalajara

Autenticación de Usuarios usando Biométricas del Ratón

**CINVESTAV
IPN
ADQUISICION
DE LIBROS**

Tesis que presenta:
José Octavio Gutiérrez García

para obtener el grado de:
Maestro en Ciencias

en la especialidad de:
Ingeniería Eléctrica

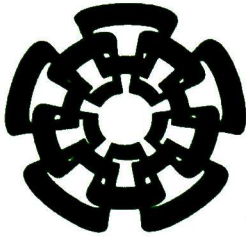
Directores de Tesis
Dr. Félix Francisco Ramos Corchado
Dr. Herwig Unger

CINVESTAV I.P.N.
**SECCION DE INFORMACION
Y DOCUMENTACION**

Guadalajara, Jalisco, Septiembre del 2006.

CLASIF.: TK165.G8 .G88 2006
ADQUIS.: SSI-412
FECHA: 15-V-2007
PROCED.: Depu.-2007
\$ _____

10' 130757-1001



CINVESTAV

Centro de Investigación y de Estudios Avanzados del I.P.N.
Unidad Guadalajara

User Authentication using Mouse Biometrics

A thesis presented by:
José Octavio Gutiérrez García

to obtain the degree of:
Master in Science

in the subject of:
Electrical Engineering

Thesis Advisors:
Dr. Félix Francisco Ramos Corchado
Dr. Herwig Unger

Guadalajara, Jalisco, September 2006.

Autenticación de Usuarios usando Biométricas del Ratón

**Tesis de Maestría en Ciencias
Ingeniería Eléctrica**

Por:

José Octavio Gutiérrez García

Ingeniero en Computación

Universidad de Guadalajara 1998-2002

Becario del CONACYT, expediente no. 191493

Directores de Tesis

Dr. Félix Francisco Ramos Corchado

Dr. Herwig Unger

CINVESTAV del IPN Unidad Guadalajara, Septiembre del 2006.

User Authentication using Mouse Biometrics

**Master of Science Thesis
In Electrical Engineering**

By:

José Octavio Gutiérrez García

Engineer in Computer Science

Universidad de Guadalajara 1998-2002

Scholarship granted by CONACYT, No. 191493

Thesis Advisors:

Dr. Félix Francisco Ramos Corchado

Dr. Herwig Unger

CINVESTAV del IPN Unidad Guadalajara, September, 2006.

AUTENTICACIÓN DE USUARIOS USANDO BIOMÉTRICAS DEL RATÓN

Esta tesis está enfocada en los ataques internos, particularmente en el enmascaramiento de usuarios. En este trabajo se pretende autenticar usuarios una vez que han iniciado sesión. Haciendo esto se refuerzan las políticas de seguridad para conceder acceso a los sistemas.

Para nuestro propósito de reducir los ataques internos, seleccionamos el dispositivo del ratón como fuente de datos, tomando en cuenta eventos del clic y los movimientos del ratón, incluyendo el código de prueba de golpe, el cual indica el componente de la ventana sobre el cual se encuentra posicionado el cursor del ratón. Una vez reunida toda la información del ratón, se aplica un procedimiento estadístico para construir la conducta del usuario; esta conducta nos ayuda a re-autenticar al usuario cada determinado periodo de tiempo, con un costo computacional bajo, cubriendo escenarios de ataques tales como: computadoras no atendidas con sesiones abiertas y el robo de contraseñas. Construyendo la conducta del usuario con información del ratón evitamos el uso de equipo costoso, y de cierta manera los datos del ratón son fácilmente obtenidos a través de la interacción del usuario con el sistema, además estamos seguros de que la conducta del usuario obtenida del ratón no puede ser imitada por otros usuarios.

USER AUTHENTICATION USING MOUSE BIOMETRICS

This thesis is focused on the internal attacks, particularly on the impersonation of users by other users. In this work we pretend to authenticate users once they have logged in. Doing this we reinforce the security policies to grant access to the systems.

For our purpose of reducing the internal attacks, we selected the mouse device as data source, taking into consideration click events and mouse movements, including the hit test code that gives information about what part of the window the mouse cursor is currently over. Once gathered all the mouse information, a statistical procedure is applied in order to build the user behaviour; this behaviour help us to re-authenticate the user every determined period of time, with a low computational cost, covering attack scenarios like non-attended computer with an open session and stolen passwords. By building the user behaviour with mouse information, the use of expensive hardware is avoided, and in certain way the mouse data is easily obtained through the user interaction with the pc, besides we are assured that mouse-based behaviour of the user can't be imitated by other users.

Acknowledgments

*I would like to express my gratitude to
my wife Carmen Delia Alcaraz Jaime
for being my main support and motivation.*

*I thank my parents
Eva García Raya and Eligio Gutiérrez Montaña
for being a great example to follow.*

*I thank my advisors
Félix Francisco Ramos Corchado and Herwig Unger
for all their advices and guidance.*

*I also thank to the CONACYT
for the scholarship granted.*

INDEX

1. INTRODUCTION.....	1
1.1. Problem description.....	2
1.2. Thesis objective.....	2
2. AUTHENTICATION APPROACHES.....	4
2.1. Introduction.....	5
2.1.1. Security Threats.....	5
2.1.2. External Attackers.....	5
2.1.3. Internal Attackers.....	5
2.1.4. Intrusion Detection.....	6
2.1.5. Intrusion Detection Systems.....	6
2.1.6. Authentication Models.....	7
2.2. Active Authentication Approaches.....	8
2.2.1. Password.....	8
2.2.2. Keystroke Dynamics.....	8
2.2.3. Physical/Biological Biometrics.....	8
2.2.4. Mouse Signature.....	9
2.2.5. Mouse Movements.....	9
2.3. Passive Authentication Approaches.....	9
2.3.1. Network User Profiling.....	10
2.3.2. Audit Data.....	10
2.3.3. System Usage.....	11
2.3.4. Command Usage.....	11
2.3.5. Content in the Title Bar.....	11
2.3.6. Keystroke Dynamics.....	12
2.3.7. Mouse Dynamics.....	12
2.4. Discussion.....	13
3. MOUSE DATA ANALYSIS.....	15
3.1. Introduction.....	16
3.2. Feature Identification.....	16
3.2.1. Mouse data collection.....	16

3.2.2. Mouse position.....	16
3.2.3. Hit test code percentages.....	18
3.2.4. Mouse clicks.....	19
3.2.5. Movement distance.....	20
3.2.6. Wavering.....	21
3.2.7. Feature vector.....	22
3.3. Feature vector from an application perspective.....	22
3.4. Definition of the user behaviour.....	26
3.5. Conclusions.....	26
4. DETECTOR ARCHITECTURE.....	28
4.1. Introduction.....	29
4.2. Main architecture.....	29
4.2.1. Mouse sensor.....	30
4.2.2. Feature extraction unit.....	30
4.2.3. Behaviour construction unit.....	30
4.2.4. Inhibitor/Reporting unit.....	34
4.2.5. Evaluator unit.....	34
4.2.6. Behaviour update unit.....	35
4.2.7. Database.....	36
4.3. Conclusions.....	37
5. EMPIRICAL EVALUATION AND RESULTS.....	38
5.1. Introduction.....	39
5.2. Decision Tree Classifier Experiment.....	39
5.2.1. Algorithm description.....	40
5.2.2. Experiment design.....	40
5.2.3. Results.....	40
5.3. One-Class Statistic Classifier Experiment.....	42
5.3.1. Algorithm description.....	42
5.3.2. Experiment design.....	42
5.3.3. Results.....	42
5.4. Conclusions.....	43

6. CONCLUSIONS AND FUTURE WORK.....	45
6.1. Related work comparison.....	46
6.2. Conclusions.....	47
6.3. Future work.....	47
 REFERENCES.....	 49

Chapter 1 Introduction

Summary

It will be explained the problem description, the thesis objective, as well as the followed process to achieve it.

1.1. Problem description

In the recent years security has become one of the most important issues for computer systems, since it is necessary to maintain privacy, integrity and availability of the computational resources. In order to reach this, a security system has to involve three concepts: prevention, detection and reaction. This work is focused on the detection phase, particularly on one of the most complicated kind of attack, the masquerade attack [SCHONLAU01], since most of the cases the attackers are employees of high levels of trust [GRANDVAUX04]; the masquerade attack by itself is a serious form of computer abuse, moreover it can be used as an initial point to perpetrate other attacks, therefore methods for user authentication should be reinforced.

The methods used for verifying the authenticity of the users have several disadvantages such as the passwords which can be easily violated by brutal force attacks or guessed in the case of weak passwords; other security devices that perform similar functions are the smart cards or USB keys, which have the disadvantage of being forgotten by the users or they can be stolen for later intrusions; the biometric approach based on physical or biological characteristics is usually expensive and enterprises need to buy specialized hardware and software, which frequently have a high cost, examples of this kind of biometrics are: DNA [CUNNINGHAM01], Infrared thermogram (facial or hand vein) [SOCOLINSKY03], characteristic odor [KOROTKAYA03], ear shape [SCHNEIER04], hand geometry [WONG03], finger prints [MAIO02], face recognition [PHILIPS02], iris recognition [DAUGMAN04], etc. (other biometrics can be found in [DELAC04] and [JAIN04]); another kind of biometrics is the behavioural that is based on behaviour characteristics of the persons, which are very complex to model and distinguish among all the users; examples of this behavioural biometric are the signature [GUPTA97], gait [NIXON04], keystroke dynamics [ENZHE04] and mouse movements [PUSARA04], among others.

1.2. Thesis objective

The main objective of this work is to develop a method to authenticate the users, through the free interaction of the user with the computer systems in order to reduce the masquerade attack; therefore this method should be able to re-authenticate the users every determined period of time with the following characteristics:

It should run in the background to provide a transparent authentication.

It should be enough light with the intention of avoiding interference with the user activities.

The data source to identify the user has to be irrelevant in order to protect the user privacy and it also ought to be easily acquired.

It must consider the changes of the user behaviour caused by variations on the environment.

Briefly it will be described the followed process to accomplish the user authentication based on the characteristic described above, and that will be detailed in the next chapters:

A revision of the user authentication methods and a classification by the data source used to profile the user were done. After a detailed analysis of these data sources, the mouse device was selected as data source; next mouse data was collected in order to be analyzed with the intention of building the user behaviour; afterwards an architecture of a user authentication system was implemented taking into consideration a feedback loop that provides updated user behaviours; finally an evaluation of our approach was made, that it will be discussed in the last chapters.

The thesis is structured as follows: chapter 2 discusses the state of the art; chapter 3 presents the analysis of the mouse data for the construction of the user behaviour; chapter 4 refers to the system architecture; chapter 5 presents an empirical evaluation of the proposal, and finally in the chapter 6 there are some concluding remarks, a comparison with related work, and the future work.

Chapter 2 Authentication Approaches

Summary

The purpose of this chapter is to give a description of the state of the art of user authentication; to reach this, it will be explained and discussed several researches which attempt to implement the user authentication using the active model or the passive model.

2.1. Introduction

Given the importance nowadays of the security issues, it has to be characterized what can be considered as security threat and how many kinds of attackers are, in order to define strategies to protect our systems.

2.1.1. Security Threats

The attackers are responsible for security threats. A threat is anything that can disrupt the operation, functioning, integrity, availability and dependability of a system [CANAVAN00].

There are two general categories where attackers can be grouped [GRANDVAUX04]:

- External attackers.
- Internal attackers.

2.1.2. External Attackers

An external attacker can be seen as someone who tries to bypass the firewall of a company or to exploit vulnerabilities in the web server; however an external attacker can also make use of the social engineering. The social engineering is a set of methods to manipulate people from outside the organization through the telephone, email, etc., based on how is structured the request; therefore nowadays is crucial for the companies to establish a set of security policies to prevent social engineering. Details of social engineering awareness can be found in [GRANDVAUX04].

2.1.3. Internal Attackers

Internal attackers are more difficult to detect, because in most cases the internal attackers are employees of high levels of trust [GRANDVAUX04] and it may be hard to distinguish whether they are using company's resources legitimately or illegitimately. Given the complexity of the problem, one way to solve it or at least to attenuate this problem, is authenticating the users, reducing the attacks such as: a user masquerading another user, in cases of disregarded computers with open sessions, stolen passwords, brutal force attacks to determine the passwords, or another different type of attacks which could be perpetrated once the attacker obtains access to the system.

2.1.4. Intrusion Detection

With the purpose of defining “intrusion detection” it has to be defined what an intrusion is, in general terms an “intrusion” can be defined as an unauthorized attempt or achievement to access, alter, render unavailable, destroy information on a system or the system itself. Therefore “intrusion detection” is the art of detecting unauthorized, inappropriate, or anomalous activities.

2.1.5 Intrusion Detection Systems

There are many security solutions that can be considered as prevention such as: antivirus [NORTON06], cryptography [RIVEST78], firewalls [TMF06], etc., however there's generally no detection, and there's almost never any reaction against intruders, due to that, a relatively new technology has emerged called Intrusion Detection Systems (IDS). These IDSs are the last line of defense against attacks, behind network security and secure program design. One essential feature of IDSs is the ability to detect attacks either internal or external and react in opposition to them.

According to [HAKAN00] Intrusion Detection Systems can be classified in three categories:

- **Network based IDS:** runs on the network and monitors all network activity. It usually employs a dedicated network server for monitoring and analyzing all traffic in real-time as it travels across the network, but this kind of IDS usually suffers from scalability problems and tends to be a bottle neck when encrypted communication is used. Some examples of this category are GrDIS [STANIFORD96], Bro [BRO06], and NetRanger [CISCO06].
- **Host based IDS:** resides on the host analyzing data from system logs and is capable of automatically monitoring and denying services if suspicious activity is detected; however most of the network based attacks can't be detected. An implementation can be found in Ref. [DITYAN03].
- **Application based IDS:** this is based on monitoring events from particular applications; it usually gathers data from system calls made by applications. Examples of application based IDS are [GHOSH99], [KUMAR04], and [MARIN01].

The one more suitable for user authentication can be the host based approach since it resides on the host allowing the extraction of the data to profile the users; consequently it is convenient to use the anomaly detection approach, which builds a normal behaviour, and anything that significantly deviates from the “normal model” arise an intrusion alarm; works like [LANE97] and [GHOSH99] make use of the anomaly detection approach to detect anomalous activities.

The performance of the IDSs is measured by the anomaly detection rates which are described below:

- The False Positive Rate (FPR): total number of false positives divided by the total number of valid instances.
- The False Negative Rate (FNR): total number of false negatives divided by the total number of real attacks.

2.1.6. Authentication Models

In order to authenticate the users there are two different models:

- **Active Authentication:** it consists in one time authentication, usually in the beginning of the sessions where the user must follow certain predefined steps in order to obtain a specific pattern to be compared with previous patterns of the current user. Examples of this approach are: passwords, keystrokes dynamics [MONROSE00], and physical/biological biometrics [DELAC04].
- **Passive Authentication:** the passive authentication model can be considered as a special kind of IDS, since detects when an intruder has logged in; this is done in the following way: the user freely interacts with the system and every determined period of time the user is re-authenticated without any predefined steps. This kind of authentication permits to hide the re-authentication process and covers more attacks scenarios. The passive authentication can be implemented using several data sources such as: application/command usage [LANE97], event log data [DENNING87], content in the title bar [GOLDRING03], mouse dynamics [PUSARA04], among others.

2.2. Active Authentication Approaches.

This kind of authentication is probably the most broadly used. In the next sections will be described some implementations of this approach.

2.2.1. Password

One of the active authentication methods mostly used is the password, however this is the weakest manner of user authentication, for the reason that the majority of the users select weak passwords like important dates, names or dictionary words that can be easily guessed or violated by brutal force attacks; nevertheless the password approach can be reinforced with the use of keystroke dynamics, as we'll see in the next section.

2.2.2. Keystroke Dynamics

It has been proved that every person has a unique typing rhythm [GAINES80]. With the aim of identifying these unique characteristics there are many metrics which are used such as: the key press duration, keystroke speed, even whether the user has pressed a key without have released the prior key, another feature utilized is the pressure applied to a key, when this is supported by the keyboard.

The keystroke dynamics is usually utilized as a reinforcement mechanism of the password like in [BIOPASSWORD06], nevertheless the majority of the experiments presented in the papers, the user's passwords are quite simple [JOO06] enabling the users the quick typing of them; however the establishment of good passwords normally involves capital letters, numbers, even punctuation; then when a strong password is assigned to the users, it begins a learning process, and this learning process is at the same time of the enrollment phase in the keystroke dynamics system; after that, the users begin to improve their typing speed of the password and therefore they modify their patterns, getting false rejection in the authentication system. More detailed information can be found in [ENZHE04].

2.2.3. Physical/Biological Biometrics

A physical/biological biometric system is based on pattern-recognition of some characteristics that persons possess, which make them distinguishable among all the persons. This biometric system is accompanied of special devices which extract the features of each person, and then the features are stored in a database for its later comparison in the verification process. There are many biological/physical biometrics such

as: characteristic odor [KOROTKAYA03], ear shape [SCHNEIER04], hand geometry [WONG03], finger print [MAIO02], face recognition [PHILIPS02], iris recognition [DAUGMAN04], DNA [CUNNINGHAM01], etc., most of them are very precise; however the necessary hardware generally has a high cost, and in some cases not all the users are allowed to register because they suffered an accident in their body, modifying their features or disabling the possibility of the enrollment; in other cases like the DNA which is high distinctive, nowadays is not a quick process. A more detailed description of the physical biometrics as well as a comparison of the different physical biometrics can be found in [DELAC04 and ANIL04].

2.2.4. Mouse Signature

The use of the mouse-based signature has its basis on the premise that the hand signature has been well accepted in government, legal and commercial processes as a method of verification; therefore a mouse-based signature could be well accepted. One of the drawbacks of this approach is the acceptance of the users, because this technique is relatively slow compared with keystroke dynamics, and also some users could present some difficulties writing with the mouse, that is not frequently done. Studies presented in [HANSHENG05, ROSS03 and KALEZHI04] give a complete description of this approach.

2.2.5. Mouse Movements

Studies made in [GAMBOA03 and GAMBOA04] establish a user authentication process by completing a matching memory game using the mouse device, while the mouse movements are being monitored and analyzed. The features extracted are the jitter, pauses in motion, velocity, angles, as well as mean and standard deviation to get a 63 dimensional feature vector. Given the characteristics of the game, the users are invited to follow a set of points or cards where they must click in certain order, implying the active authentication model. The results obtained in their experiments were very good, however the time required in order to authenticate the users was large, resulting in a non-practical solution for the user authentication at least in the active authentication model.

2.3. Passive Authentication Approaches.

The active authentication precedes the passive authentication, being the passive authentication the most complicated since is established in a completely dynamical environment. Most of the implementations of passive authentication are based in the user

profiling, taking into account various data sources of the user interaction. In this section will be explained some researches of the passive authentication method, making emphasis in the mouse dynamics, since the user authentication via the mouse dynamics is part of our proposal.

2.3.1. Network User Profiling

The users can be profiled through their network activities, which involve the used application (web, ftp, telnet, etc.), accessed servers, and also the amount of transferred data; all this information is analyzed to build the user profile. Researches such as [MARCHETTE99] use a statistical method to profile the users, however the behaviour of the users based on accessed servers is highly unstable, because it highly depends of the user activities and not in the characteristics of the user; another drawback could be that the users are not forced to use the network in order to be authenticated, for the reason that not all their actions involve the network, disabling the user authentication process.

2.3.2. Audit Data

The first intrusion detection systems were based on audit data generated by the users [DENNING87], in this approach are analyzed the login, session activity, command and program execution and the file access activity; this information is obtained from the system's log records, then a model is constructed, that contains the behaviour of subjects with respect to objects in terms of statistical models and metrics such as event counters, interval timers, and resource measures; the statistical models utilized were: the operational, mean and standard deviation, multivariate, markov process, and time series. A more recent work is presented [YE01], that basically used the same data sources than [DENNING87] but introducing the frequency property, the duration property, and the ordering property of the events recorded in the system's audit data; using a probabilistic method they could detect with more precision intrusion detection behaviours. Unfortunately, given the complexity of the current operative systems, the majority of the system's audit data are not directly generated by the user, this is that it can establish a normal behaviour of the system, but it can't define the behaviour of the users, at least not enough to authenticate them.

2.3.3. System Usage

In order to measure the system usage there is a variety of features that can be extracted such as: process numbers, memory and disk information, paging rates, disk space free, etc. The work presented in [BURGESS01] represents this data as a time series, measuring its entropy which is a numerical indication of the variations of the signal and it also suggests the form of the distribution of the data about the mean. They also showed that the entropy always increases until a steady state is reached. The periodicity of the data taken in their experiments was daily and weekly, being this the main drawback of this approach, since the required time to authenticate the users is extremely large; therefore to measure the system normality can't be efficiently used to authenticate the users.

2.3.4. Command Usage

The command usage approach argues that there is a set of commands/aliases which are commonly used by the users during their normal activities; monitoring these command executions and their order, the user behaviour can be built, once constructed it can be determined if an intruder is trying to impersonate a legitimate user. Many researches have proved the consistency of the command usage [SCHONLAU01, MAXION04]. The command usage was probably one of the approaches more utilized for profiling users, since many of the operative systems were command line based, although the command-line is still utilized by the users, most of the operative systems are migrating or already have graphical user interfaces to configure and use the system, becoming the command usage an outdated approach.

2.3.5. Content in the Title Bar

The content in the title bar approach is the adaptation that suffers the command usage, since almost all the operative system already have migrated to graphical user interfaces. The use of approaches like application usage don't provide enough information to authenticate users, then analyzing the content in the windows title bar gives more details of the user. The content in the title bar approach is broadly studied in [GOLDRING03]. Some of the features extracted from the title bar are the following:

- Time between windows.
- Time between new windows.
- Number of open windows at the same time.

- Amount of time of the open windows.
- Number of words in the title bar.

Unfortunately the performance of this approach hasn't been widely proved yet.

2.3.6. Keystroke Dynamics

The keystroke dynamics approach can be applied in both models active [ENZHE04] or passive [AHMED05]; however for the passive model the construction of digraphs/trigraphs that hold the relations between consecutive letters typed is necessary; therefore when free text is analyzed the size of the digraphs/trigraphs tends to be very large, moreover the amount of data necessary to generate the graphs is also considerable large, with a high computational cost of the graphs generations. In [AHMED05] they used a key oriented neural network to simulate user keystroke dynamics with reference to other digraphs/trigraphs previously generated, approximating the values of different digraphs/trigraphs based in the locations of the keys with reference to each other, aiming to speed up the enrollment phase of the users; this reduces the number of data necessary but increases the computational cost. Other work that studies the keystroke dynamics using free text is described in [HOCKET04] here the authors take into consideration the pressure applied to the keys. The big disadvantage of this approach is the necessity of a keyboard that supports to measure the key pressure. The above works don't consider that the variability of the features is higher than in the active model, because during a user's sessions the speed of the typing can be easily affected by externals and internal distractions, externals such as when the user is holding a cup of coffee with one hand and typing with the other or typing in different positions than the usual, and internal such as when the user is tired after a long journey, among other distractions.

2.3.7. Mouse Dynamics

The same as keystroke dynamics, mouse dynamics can be applied in both models active and passive. In the section about mouse dynamics of the work presented in [HOCKET04], the authors studied whether the users can be authenticated based on their usual interactions with the computer mouse using a statistical approach. According to their results the information provided by the mouse wasn't enough to authenticate the users. Other work that attempts to establish an authentication process using the mouse dynamics is described in [HASHIA05], the strategy followed by them is described next: first, they

group the mouse coordinates recorded from the user interaction into dense regions; second, the regions are covered by a convex polygon being obtained all the regions where the cursor mouse moves most of the time, calling them *states*; finally they find the probability of the transitions from one state to other, using Hidden Markov Models; however they couldn't perform a correct authentication process. In the study described in [PUSARA04], they extract raw features such as: distance, angle, time and speed; and extracted features such as: mean, standard deviation and third moment of all the raw features; they also consider the mouse wheel, No-client moves, single click and double click; gathering all these characteristics they obtained 111 features; then they apply a supervised learning method (decision tree classifier) to discriminate among k users, using a smoothing filter overlapping the frequency of the mouse data points with the objective of reducing the false alarms; however given their results, they pointed out that analyzing mouse movements alone is not sufficient for a stand alone user re-authentication system. Following a similar approach is [AHMED05], here the mouse actions are classified in four categories: mouse movement, drag and drop, point and click, and silence; then for each category is extracted features such as: mean and standard deviation; once that all the features were extracted, a statistical method is applied to profile the users, combining the mouse dynamics with the keystrokes dynamics. The results of their experiments were good; nonetheless the required amount of data to enroll the users was very large, since their approach is based on a key oriented neural network, which increases the required time in the enrollment phase and the computational cost in the training phase, resulting in an impractical registration of new users in the system.

2.4. Discussion

Both the active and the passive authentication model are required to implement a user authentication system, even with all the security measures we can not stop the internal attacks at all, but with a complete authentication system we can decrease its frequency. It has been already analyzed the advantages and disadvantages of the approaches of the active and passive model authentication; however one important element that most of the approaches don't consider is the user privacy, here is where many of the previously described approaches fail; network data, system log data, content in the title bar and keystroke dynamics are examples of data sources which infringe the user's privacy. Then it has to be considered the privacy issues, therefore the selection of a data source such as mouse data that protect the user privacy is a good option. The majority of the passive

authentication approaches are behaviour-based and don't contemplate the variations of the user behaviour through the time or by lights modifications in the environment, then a feedback mechanism is required, with the function of updating the user behaviour, maintaining as low as it is possible the anomaly detection rates. Another common drawback is the large amount of data required to profile the users increasing the computational cost and the enrollment time.

Chapter 3 Mouse Data Analysis

Summary

This chapter describes the followed process to identify the features that model the user behaviour, recognizing the permanency of the features through different sessions and also the variability of the same feature among the users.

3.1. Introduction

The analyzed data was taken from two sessions, participating 14 users; the first session had as main objective to collect data of the users from only one application with the intention of identifying features that will help to model the user behaviour; the second session had as objective to collect data coming from different applications to observe possible differences of the user behaviour according to the application; as a result of the analysis 5 categories of mouse data emerged, which were distinctive for each user and permanent through the user sessions.

3.2. Feature Identification

The followed process to identify the features was merely graphical and statistical using basic statistics such as mean, standard deviation and percentages.

3.2.1. Mouse data collection

In the first collection session the users used the same mouse device, with the aim of eliminating any kind of noise that could alter the user behaviour; afterwards it was asked to the users to use Microsoft Windows and Internet Explorer for the time that takes to record 20,000 mouse points every 50 milliseconds, recording mouse points only when the user moved the mouse. The time interval corresponds to the balance between significance of the mouse data and the computational cost produced by its computation. In the second collection session the users were working on their normal activities recording mouse points from different applications which demand different efforts to the users and therefore changes in their behaviour appeared. In both sessions it was also stored all the mouse events generated by the mouse wheels and mouse clicks.

3.2.2. Mouse position

According to the information extracted from the collected data, it could identify the permanency of the mean and the standard deviation of the mouse coordinates. As we can see in the figure 3.1, there is a comparison between two users in five sessions, it also illustrates a strong permanency of the features of the user 2; nevertheless the user 1 presents more variations in his values, giving evidence for believing that not all the users have a consistency in all the features, being this a fundamental part of the extraction of the user behaviour, where some users are consistent in some features, others don't.

In addition we develop a new strategy to help the user authentication based on the identification of the regions where the user makes more mouse movements; as is shown in the fig. 3.2 the distribution of the mouse data points is extremely different among all the users, it could also identify the permanency of these distributions through the user sessions, from here it was extracted 48 features, dividing the mouse data point distribution in a 6x8 grid as is shown in the fig. 3.3, afterwards is computed the percentage of the mouse data points per cell.

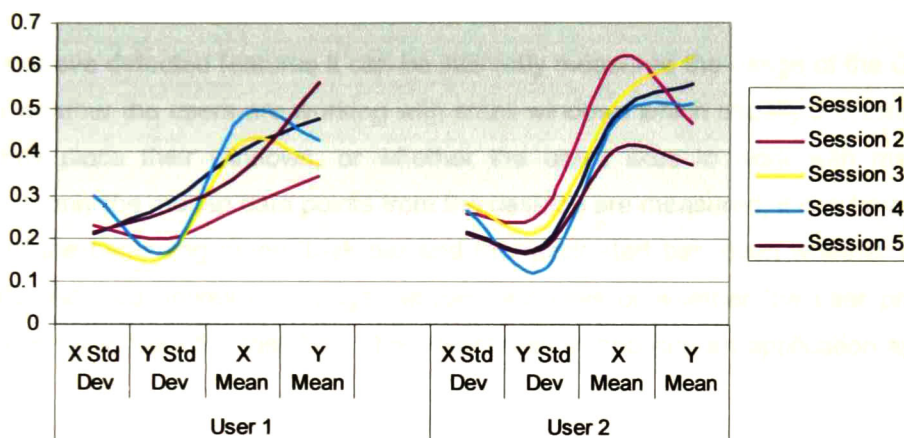


Figure 3.1. Mean and standard deviation of the mouse coordinates

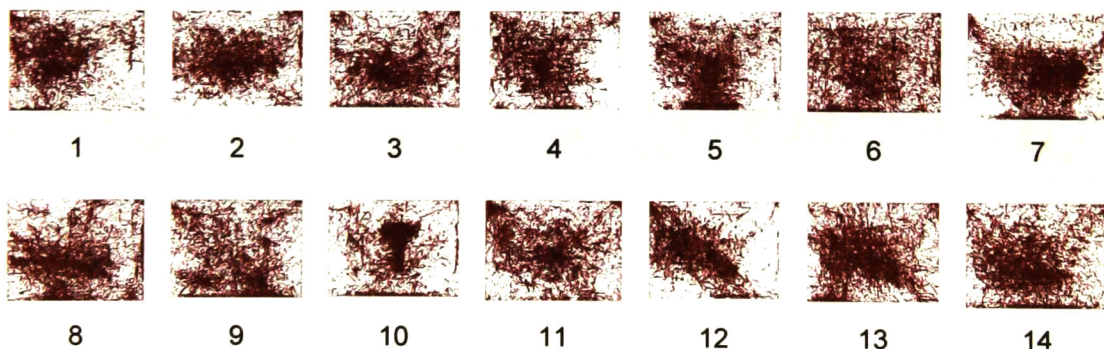


Figure 3.2. Mouse point distribution of each user.

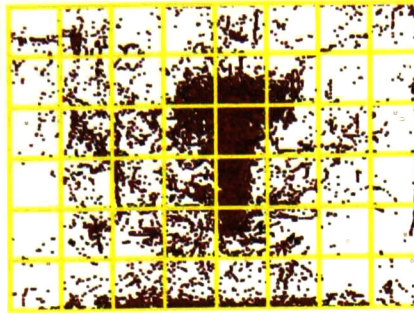


Figure 3.3. Mouse data point distribution divided by a 6x8 grid.

From the above detected features it can be indirectly measured the usage of the GUIs, for example whether the users are working with small windows which usually don't maximize, where they place their windows, or whether the users likes to work with maximized windows. If only the mouse data points from the desktop are measured, it can be extracted the use of the beginning menu, task bar and the quick start bar, even whether the user utilizes the Alt+Tab hotkey to change between windows or whether the user prefers to select the window from the task bar. The advantages of applying an application approach are evident.

3.2.3. Hit test code percentages

The hit test code provide important information of the usage of the GUIs, for example it gives information about what part of the window the mouse cursor is currently over. There are 23 hit test codes but only 12 of them are enough frequent to provide significant information. The hit test codes taken into account are described in the table 3.1:

Code number	Description
1	Mouse cursor in the border of a window that does not have a sizing border.
5	In a title bar.
6	In a client area.
7	In a close button.
8	On the screen background or on a dividing line between windows (same as HTNOWHERE, except that the DefWindowProc function produces a system beep to indicate an error.
11	In a horizontal scroll bar.

13	In a menu.
14	In maximize button.
15	In minimize button.
16	On the screen background or on a dividing line between windows.
19	In the upper horizontal border of a window.
23	In the vertical scroll bar.

Table 3.1. Considered hit test codes.

Taking into account the hit test codes are measured the usage of the scroll bars, system menu, minimize, maximize, restore actions, even whether the user prefers to utilize the close button from the system menu or the Alt-F4 hotkey. The figure 3.4 shows the averages values of the hit test codes of five sessions of two users, we can easily observe two values which are pretty different, the case of the HTC 5 and HTC 16, although the other values are pretty similar and the majority of the values are close to zero, being evidence of the use of the hot-keys.

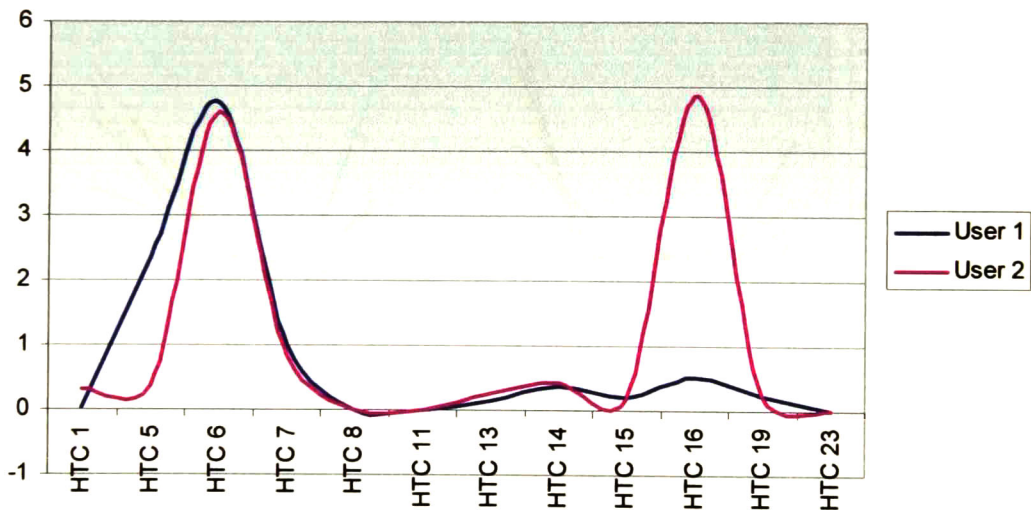


Figure 3.4. Hit test codes comparison

3.2.4. Mouse clicks

The mouse clicks can be classified as single or double, in this work only single clicks are considered, since the double clicks are a combination of the single clicks; now a single click can be performed using different buttons (left and right) and it has normally different

duration depending on the button pressed and it's maintained for each user through different sessions. Given the variety of the clicks duration, the mouse clicks were divided in 5 groups of click duration for each button, from 0 to 100, 100 to 200, 200 to 300, 300 to 400, and bigger than 400 ms. The majority of the applications have the option of popup menus, and the use of these menus is another feature to distinguish the users; the users normally invoke this menu with the right click, then the percentage of the use of the right click should be measured. In the figure 3.5 we can see the left click duration of two users, here the user 2 tends to do quick clicks and the user 1 doesn't make clicks from 200 to 400 ms; these characteristics are maintained through 5 sessions, however the fourth session is abnormal in both users. This could be due to external and internal influences of the environment which could modify the user behaviour, then sometimes is normal to have this variations. It was also noticed that the duration of the mouse clicks varies depending on the activity performed by the user, which is linked to the application; at the end of this chapter a comparison of the use of different applications by the same user will be presented.

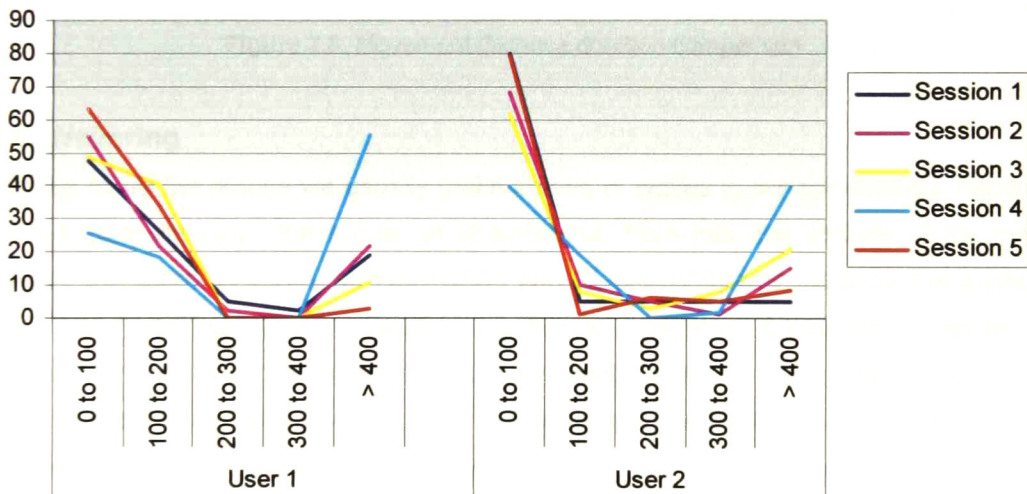


Figure 3.5. Left click duration comparison

3.2.5. Movement distance

The mean of the movement distance is maintained through different sessions and it is also different for each user; however the same as the click duration, the movement distance could be grouped in 4 categories according to its distance, from 0 to 5, 5 to 20, 20 to 100 and bigger than 100 pixels; similar to the other features the movement distance is strongly

linked to the application, since some GUIs have tool bars with small buttons and others have big buttons, resulting in smaller or bigger mouse movements depending on the application. The figure 3.6 shows the consistency of the movement distance through the sessions and it also shows the differences between the users.

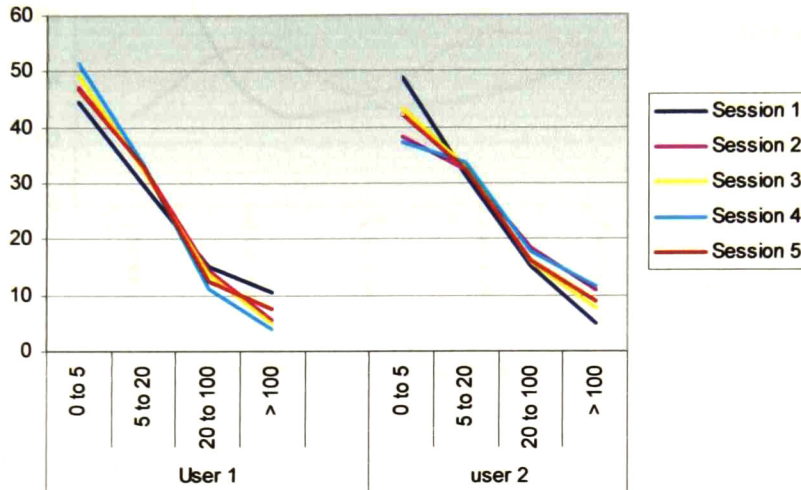


Figure 3.6. Movement distance duration comparison

3.2.6. Wavering

When we move the mouse we usually make curves or waves in our movements, because we don't normally move the mouse in straight line, from this, the altitude of the curve (wave) is extracted. Contrary to [HASHIA05 and PUSARA04] who consider the angle of the movement taking into consideration positive and negative angles according to the direction of the wave, however according to the analysis to the data of the 14 users the angle of the curve is maintained in both directions and the percentages of the negative and positives curves are completely arbitrary, therefore only the altitude of curve is considered. In the figure 3.7 we could see that the user 2 does more wavering in his movements than the user one, with exception of the session one, which has a high variation.

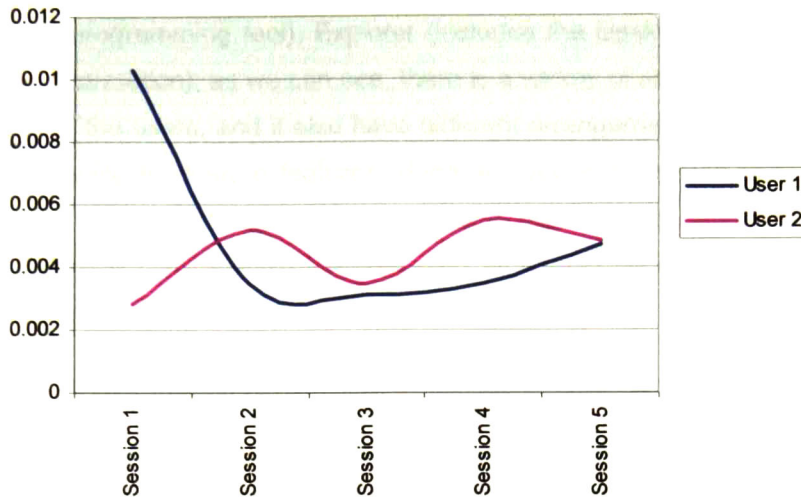


Figure 3.7. Wavering comparison

3.2.7. Feature vector

The followed process to detect the above feature categories was merely statistical and graphical in order to identify relations between the features and their permanency through different sessions; as a result of this process, a set of 81 features were identified, which were the only significant ones according to our study. A set of these feature vectors will be part of the user behaviour, one feature vector per application since the behaviour of the user changes according to it; this will be discussed in the next section.

In the next chapters, the correct selection of the 81 features will be demonstrated by making an empirical evaluation, using several window sizes (number of points) in the construction of the feature vectors demonstrating the feasibility of the extracted features and identifying the correct window size for each user.

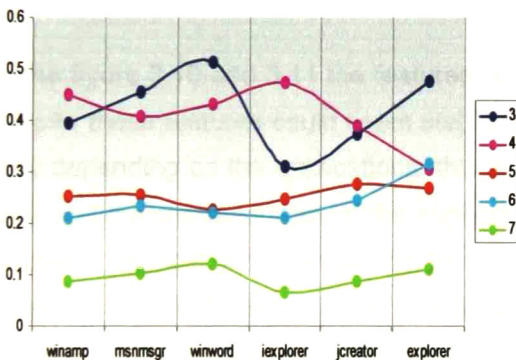
3.3. Feature vector from an application perspective

It was already shown the consistency of the features through the different sessions of the users in the past sections, nevertheless as it was mentioned the majority of the features change according to the application, to show this, the data collected in the second sessions will be analyzed; for purposes of the explanation I'll only present the analysis of the behaviour of two users in six different applications with dissimilar functions and dimensions of the window. The analyzed applications were: Winamp (media player), MSN

messenger (chat), Microsoft Word (word processor), Internet Explorer (web navigator), JCreator (a java programming tool), Explorer (includes the Desktop and all the windows which show file information); as we can see, there is a variety of applications which require different efforts of the users, and it also have different arrangement of components of the window. I'm not going to analyze features which are obviously related to the application such as mouse points per region or mean and standard deviation of the mouse coordinates, therefore I will analyze features such as hit test codes, movement distance, and click durations.

The next graphs show the values of the features with the corresponding application. The values of the features were extracted using a window size of 4000 mouse points with the objective of enhancing the differences in the behaviour of the users; in each graph are presented 5 or 4 features for making clearer graphs.

User 1



User 2

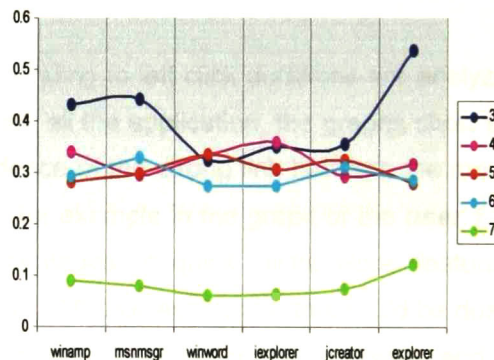


Figure 3.8. Application usage comparison 1/10

Figure 3.9. Application usage comparison 2/10

As the figures 3.8 and 3.9 show, the series of the feature 3 (mean distance between consecutive points) show big differences between the users, let's observe the value of the winword application, here the *user 1* makes bigger movements than the *user 2* nevertheless they are using the same application. The graphs clearly show that the features in some cases are very different depending on the application as example the value of the feature 3 of the application Winword and Internet Explorer of the *user 1*; however the feature 7 (movements of size bigger than 100 pixels) is highly stable in both users, though we cannot establish that all the users hold this, for example the feature 4

(Movement Size $0 < \text{Distance} \leq 5$ pixels %) is stable in the graph of the *user 2*, but this relation is not held in the graph of the *user 1*, then some users hold certain features in all the applications or some don't.

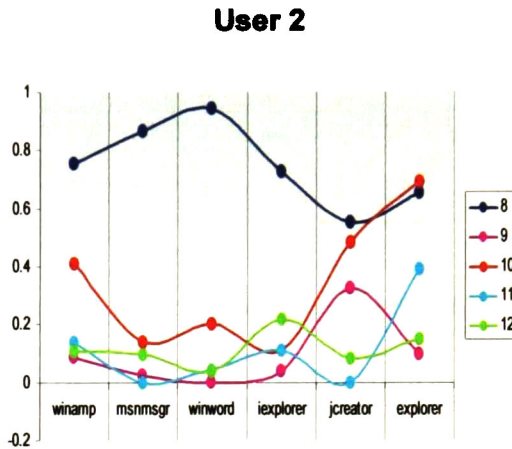
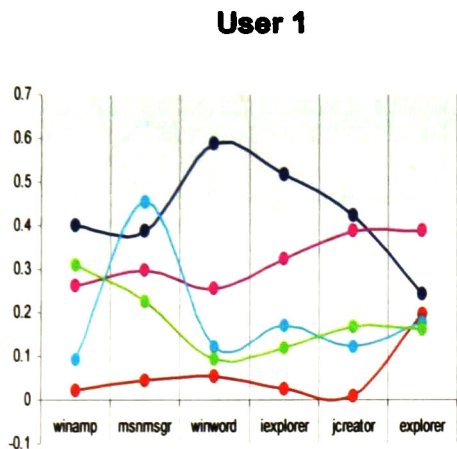


Figure 3.10. Application usage comparison 3/10 **Figure 3.11.** Application usage comparison 4/10

In the figure 3.10 and 3.11 the features corresponding to left click durations are analyzed, despite these features could seem stable through all the application, the graphs show that vary depending on the applications; this is evidence of the strong link between the mouse click and the characteristics of the applications, for example in the graph of the *user 1* the JCreator application receives the lowest percentages of quick clicks since feature 8 correspond to: Left Click Duration $0 < \text{duration} \leq 100$ milliseconds %, this could be due to a relation in the level of concentration of the user during his programming activities and the click of the mouse.

In the figures from 3.12 to 3.17 are presented the values of the remaining features per application, just to illustrate that the variations of the features in the different applications is considerable distinctive in the majority of the cases. As you will see, almost none of the features remain stables through the six applications in both users, giving more evidences of the shown changes by the users according to the application.

User 1

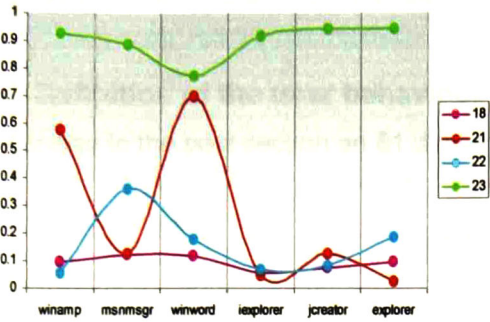


Fig. 3.12. Application usage comparison 5/10

User 2

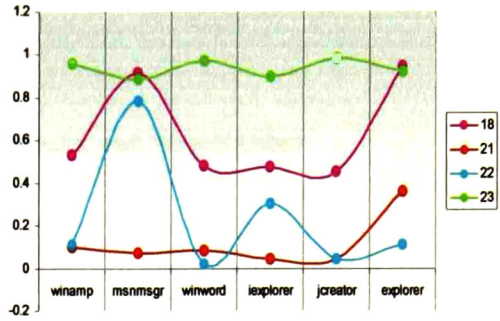


Fig. 3.13. Application usage comparison 6/10

User 1

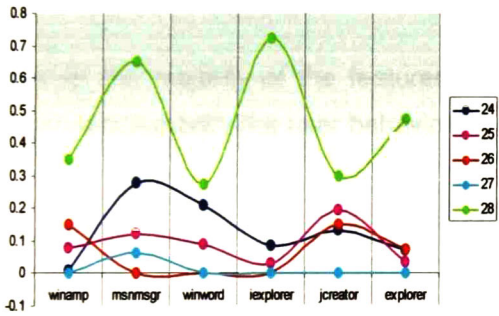


Fig. 3.14. Application usage comparison 7/10

User 2

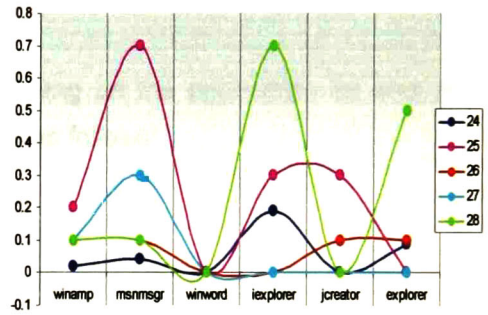


Fig. 3.15. Application usage comparison 8/10

User 1

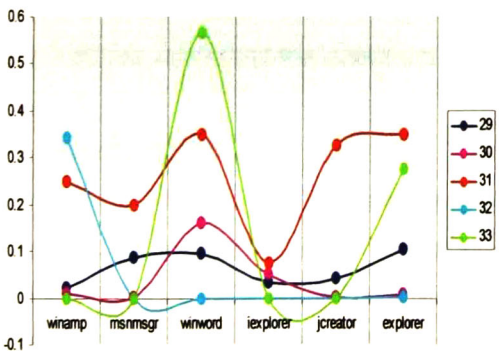


Fig. 3.16. Application usage comparison 9/10

User 2

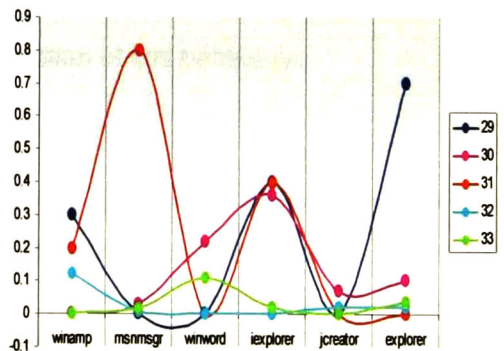


Fig. 3.17. Application usage comparison 10/10

The above graphs illustrate that the majority of the features change depending on the applications, due to the different design of the GUIs and the effort required by the users.

3.4. Definition of the user behaviour

According to the past section an 81 dimensional vector can be extracted:

$$\psi = \{\tau_1, \tau_2, \dots, \tau_{81}\}$$

Where:

- ψ is the feature vector.
- τ_i represents the feature i
for $1 \leq i \leq 81$ and $\tau_i \in \mathbb{R} \mid 0 \leq \tau_i \leq 1$

However the majority of the features vary depending on the application as was shown above; consequently the user behaviour is defined as follows:

$$\Gamma = \{\psi_1, \psi_2, \dots, \psi_n\}$$

Where:

- Γ is the user behaviour.
- n represents the number of the applications existent in the operative system.
- ψ_i is the user behaviour for a determined i application.

Later it will be explained the construction and detection of this behaviour.

3.5. Conclusions

As a result of the mouse data analysis 81 features were extracted of the interaction of the user with the mouse device. It was shown the permanency of the features through the different sessions and also the distinctiveness of each feature among the users. This chapter also confirmed that the 81 dimensional vector is widely affected by the application,

being necessary to measure all these changes, which will produce detailed user behaviours, subsequently the application approach is necessary to authenticate the users.

In later chapters will be demonstrated the correct selection of the 81 features, making an empirical evaluation, using several window sizes (number of points) in the construction of the feature vectors demonstrating the feasibility of the extracted features and identifying the correct window size for each user. It also will be explained the architecture of the user authentication system.

Chapter 4 Detector Architecture

Summary

The aim of this chapter is to present the architecture of the user authentication system, explaining in detail the functions of every component of the architecture as well as the relations among them.

4.1. Introduction

The architecture of a user authentication system should be light, since the system ought to re-authenticate the users every certain period of time or number of mouse points. In addition the processing algorithms should be fast in order to low the computational cost and avoid interfering with the normal activities of the users; additionally there should be a feedback process that update the user behaviour, due to the progressive adaptation of the users to the computer applications. The design of the architecture was provided with components with the mentioned characteristics above, and also a feedback loop was established to update the user behaviour. In the rest of the chapter will be explained the system architecture in detail.

4.2. Main architecture

The figure 4.1 depicts the system architecture that consists of six components and a database module; it also shows the existent relations among the components. It is necessary to mention the possibility of deploying this architecture as a host-based application where all the component are in the protected host, or it can be also deployed as a client-server application where the mouse sensor and the feature extraction unit are in the protected computer and the remaining components in the server. In the next subsections will be described de functions of each component and also the feedback loop conformed by the behaviour update unit and the behaviour construction unit with the intervention of the database module.

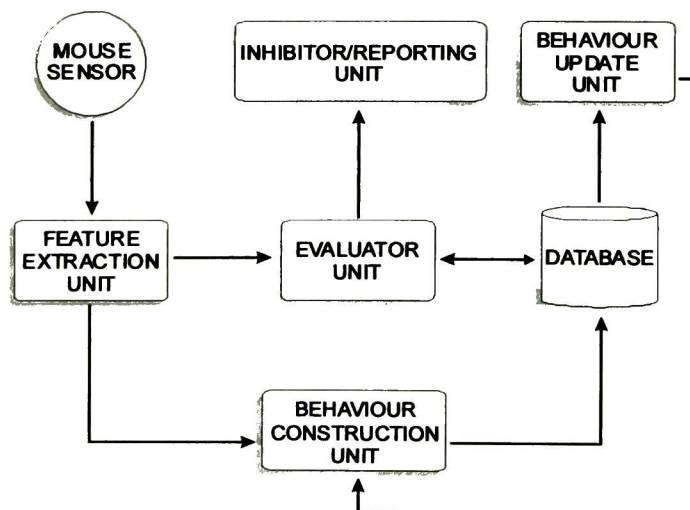


Figure 4.1. System architecture

4.2.1. Mouse sensor

The mouse sensor obtains the following information from the user interaction: cursor position, hit test codes, click events (double click, left, medium, and right click), mouse wheel events, the active application and the time; all the information is written on a file, with the following format:

```
0.630 0.633 1 0 0 0 0 0 0 0 0 0 winhlp32.exe 7971
0.629 0.633 1 0 0 0 0 0 0 0 0 0 winhlp32.exe 7972
0.629 0.633 1 1 0 0 0 0 0 0 0 0 winhlp32.exe 7973
L 0.629 0.633 94 winhlp32.exe 7973
```

Figure 4.2. Example of mouse data

The mouse sensor participates in the enrollment phase as well as the verification phase. The main objective of the enrollment phase consists in the registration of the user in the system, and the verification phase consists in a continuous monitoring of the user's interaction via the computer mouse with the GUIs in order to evaluate the collected data and emit a judgement of authorized or non-authorized user. The mouse sensor in both phases collect the data either to construct the user behaviour or to evaluate the collected data.

4.2.2. Feature extraction unit

As well as the mouse sensor, the feature extraction unit participates in both phases, the function of the unit is to extract from the raw data the 81 features previously identified in the data analysis. Basically it makes use of the basic statistics such as the mean, standard deviation and percentages.

4.2.3. Behaviour construction unit

The behaviour of each user is unique and complex and some of them have certain characteristics that make them identifiable; these characteristics need different times or in our case different number of mouse points in order to be identifiable; therefore is necessary to make a search taking into account different windows sizes (number of data points) to create the features vectors, with the intention of identifying the users in the shortest time; once the different feature vectors were extracted from the data windows, it is randomly selected a determined number of feature vectors for each application, with the

objective of observing the users at different times of their interaction with the system, capturing variations in the behaviour. Once we have these vectors, a classifier can be trained in order to identify the user behaviour on every different application of the system. The selection of the window size is determined by evaluating the remaining feature vector of each application calculating its False Positive Rate (FPR) which is represented by the total number of false positives (valid feature vectors classified as invalids) divided by the total number of valid sessions; this rate give us the level of solidity of the determined behaviour. In this work the one-class machine learning approach is used, since it is only necessary to validate one user and everything outside of that user will be rejected as intruder. The selection of one-class machine learning corresponds to its advantages such as: less data collection, independent training, fast training and testing, and its decentralized management.

As we could see in the data analysis chapter some users maintain certain features more than others, according to this level of consistency, is the degree of the significance of the feature; however the 81 features are not equally significant for all users, for example some users maintain a constant speed of the mouse movements and some users don't, then we need to determine what features are significant for each user but taking into consideration the remaining features although with less value. The preceding idea suggested to design a one class classifier called One-Class Statistic Classifier (OCSC) which is capable of selecting the most significant features of the user, and to determine a confidence range for that features, assigning a penalty value depending on their level of significance with the objective of evaluating new cases and generating a penalty value, that will be compared with a predefined threshold to determine whether it belongs to the user or not.

The OCSC has three phases: the training phase where the user behaviour is modelled; the evaluation phase where new cases are classified as valid users or intruders, and the third phase that consists in the threshold adjustment for the evaluation phase. In this section will be explained the first and third phase, which are the phases involved in the behaviour construction.

Training phase: this phase receives as input a set of training vectors which will be used to model the user behaviour in one application, that will be represented by a set of ranges,

one range per feature of the user behaviour; next it will be described in detail the training phase:

Input:

- The set of training vectors $F = \{u_1, u_2, u_3, \dots, u_N\}$

Where:

- $u_i = \{u_{i1}, u_{i2}, u_{i3}, \dots, u_{iDim}\}$ where $u_{ij} \in \mathbb{R} \mid 0 \leq u_{ij} \leq 1$
- N : number of training vectors
- Dim : number of features of the user behaviour

The objective of this phase is to compute the set of ranges R denoted as follows:

$$R = \{\text{range}_1, \text{range}_2, \dots, \text{range}_{Dim}\}$$

Where:

- **range** is a 7-tuple ($id, m, s, p, sf, inf, sup$)
- id : Feature identifier
- m : Mean of the feature
- s : Standard deviation of the feature
- p : Penalty (value assigned to qualify inputs)
- sf : Spread factor (how spread are the values)
- inf : Inferior limit of the confidence range
- sup : Superior limit of the confidence range

Now to compute each range 4 steps have to be done for every $k = 1, 2, \dots, Dim$, one per each feature:

1. Computation of the mean and the standard deviation:

$$\text{range}.m_k = \text{Mean}\left(\sum_{i=1}^N u_{ik}\right) \&$$

$$\text{range}.s_k = \text{StdDev}\left(\sum_{i=1}^N u_{ik}\right)$$

2. Computation of the spread factor: the spread factor represents how much the data is spread in each feature; this is computed in the following way

$$\mathbf{range.sf}_k = \begin{cases} sf_k = 1000 & m_k = 0 \\ sf_k = 100(s_k)/m_k & m_k > 0 \end{cases}$$

When m_k is equal to 0, the training vectors don't provide enough information to consider the feature; therefore it is assigned a high value of spread factor to give it the lowest penalty value.

3. Assignment of the penalties: the ranges are sorted by spread factor in ascendant order, in order to assign the penalty values in an easier way; subsequently the value of the penalty is assigned in this way:

$$\mathbf{range.p}_k = \mathit{Dim-k}$$

By doing this, it is assured that features which are more constant get higher penalty values and features which are inconstant get smaller penalty values.

4. Computation of the confidence ranges:

$$\mathbf{range.inf}_k = \mathbf{range.m}_k - \mathbf{range.s}_k$$

$$\mathbf{range.sup}_k = \mathbf{range.m}_k + \mathbf{range.s}_k$$

These limits provide a confidence range so as to facilitate the evaluation of new cases. As final step we sort the R set by feature identifier in ascendant order.

Output: R – Set of ranges

The above output contains the user behaviour as well as the confidence ranges for each feature giving the basis to evaluate new cases, as it will be explained in the evaluation unit.

Threshold adjustment: the procedure to set the threshold can be separated in 3 parts:

- a) Once calculated the ranges in the training phase of the OCSC, the training cases have to be tested in the evaluation phase, in order to compute the mean of the

penalizations obtained from all the training cases. As a result is obtained the *TrainingMean*.

- b) Then the testing cases that belong to the target user have to be tested in the evaluation phase and compute the mean of the penalizations. As a result is obtained the *TestingMean*.
- c) The *TrainingMean* don't provide enough error margin to discriminate the valid user from intruders because the ranges were computed using those training cases, On the other hand the *TestingMean* represents the ideal penalty amount, however it is necessary to add some error margin, to enlarge the error margin is used the *TrainingMean*, therefore the threshold is computed as follows:

$$\mathbf{Threshold} = \mathbf{TrainingMean} + \mathbf{TestingMean}$$

The complexity of OCSC in the training phase is $\mathbf{Dim} * \mathbf{N}$ where \mathbf{Dim} is the dimension of the feature vector and \mathbf{N} is the number of training cases, and the complexity of the threshold adjustment phase is linear.

4.2.4. Inhibitor/Reporting unit

This unit can be used to inhibit an intruder closing the session for several minutes or it also can be used to send a message to the security administrator.

4.2.5. Evaluator unit

The evaluation phase has as main objective to classify new cases as belonging to the target user or as intruder, besides giving the resulting penalty value of the evaluation, in order to facilitate the threshold adjustment in the third phase of the OCSC. In this unit the second phase of the OCSC is located.

The input of this phase is the set of ranges computed in the training phase, the new case to be classified, besides the threshold, which are denoted as follows:

Input:

- Set $\mathbf{R} = \{\mathbf{range}_1, \mathbf{range}_2, \dots, \mathbf{range}_{\mathbf{Dim}}\}$

- Testing vector $\mathbf{v} = \{v_1, v_2, v_3, \dots, v_{Dim}\}$
where $v_i \in \mathbb{R} \mid 0 \leq v_i \leq 1$
- **Threshold**

Next it is verified whether the value of each feature is between the ranges; if it is, there is not any penalization, but if it isn't, there is a penalization based on its penalty value assigned and in how much it differs from the range, this is computed as follows:

Penalty =

$$\text{if } (v_k < \text{range.inf}_k) \sum_{k=1}^{Dim} \frac{(\text{range.inf}_k - v_k) * \text{range.p}_k}{(\text{range.sup}_k - \text{range.inf}_k)} \text{ or}$$

$$\text{if } (v_k > \text{range.sup}_k) \sum_{k=1}^{Dim} \frac{(v_k - \text{range.sup}_k) * \text{range.p}_k}{(\text{range.sup}_k - \text{range.inf}_k)}$$

Now if (**Penalty > Threshold**) then

Result=**Intruder** else Result=**Valid user**

Output: Result & Penalty

The **Penalty** basically represents how much the testing vector \mathbf{v} differs from the behaviour of the target user, and the **Result** gives the verdict of the evaluation according to the **Threshold**. The complexity of the evaluation phase is linear.

4.2.6. Behaviour update unit

It is well known that the obtained features from a behavioural biometric of a person change through the time; then it is essential to establish a feedback loop that adjusts the user behaviour.

The feedback loop should be implemented basing us in a record of the penalties received of the feature vectors of the legitimate user; the proposed procedure is detailed as follows:

1. Store de **training vectors** utilized to generate the feature vector ψ for each application i .

2. Store a list of feature vectors extracted from the user interaction in the verification phase, let's call them "**interaction vectors**" and also store their corresponding penalty.
3. Every certain number of authentications, it should be computed the mean of the penalizations.
4. Once computed the mean of the penalizations, it is selected the **interaction vector** that has the nearest penalty value to the mean. Doing this the user behaviour is updated with a vector that represents a normal session of the user and it also presents a small deviation of the behaviour.
5. Then the oldest element of the **training vectors** should be deleted and added the **interaction vector** selected and repeat the construction of the user behaviour.

This process should be repeated for each application i .

There isn't any study of the durability of the behaviour of a user, however if the proposed procedure is utilized every determined number of authentications, given its characteristics, it assure to select the correct feature vector that will correctly update the user behaviour, no matter how often the behaviour is updated.

One common drawback of the feedback loops is the training of the system by intruders, which slowly teach the system their behaviour; however as our approach uses a mouse-based behavioural biometric, these intruders will be detected in the first session allowing us the addition of the feedback loop without any security issue.

4.2.7. Database

In the database is stored the user behaviour, the window sizes, the training vectors which conform the user behaviour, and a record of the user authentications with their corresponding penalties, for each application.

4.3. Conclusions

The mouse sensor was calibrated with the intention of capturing the necessary amount of data to compute detailed user behaviours in order to low the computational cost in the feature extraction unit, as we have already seen it only make use of basic statistics. The behaviour construction unit with the use of the OCSC selects the most significant features of the user behaviour, allowing a meticulous evaluation of the user behaviour in the evaluator unit that uses a penalty based evaluation. The behaviour update unit probably makes the most important function in order to reduce the anomaly detection rates and to extend the validity of the user behaviour, capturing the variation of the behaviour according to the adaptation or light changes in the environment.

The whole system architecture needs low computational resources to monitor and protect systems, and it is also very flexible referring to the position of the components, either in the protected host or in a client-server approach.

Chapter 5 Empirical Evaluation and Results

Summary

This chapter presents an empirical evaluation of our approach with the aim of showing the feasibility of the 81 features already detected and it also proves that it is possible to authenticate the users using mouse data and the usage of the GUIs.

5.1. Introduction

In the previous chapters the feature extraction of the mouse data and the behaviour construction were analyzed; now in this chapter corresponds to demonstrate the correct selection of both procedures, to do this, two experiments will be explained, the first one utilizing as evaluator the Decision Tree Classifier (DTC) using a supervised learning method and the second one using the OCSC. The DTC was selected as evaluator in the first experiment for the reason that it makes clearer the classification process and therefore provides a confirmation of the correct selection of the features, and the OCSC experiment provides a more practical solution for the user authentication system.

The mouse data of the experiment was taken from the mouse data collection session of the data analysis, participating 14 users; in both experiment it was only used data from one application the Internet Explorer to avoid differences in the user behaviour in different applications as was shown in the data analysis chapter, therefore the results reached in these experiments will provide a reliable evaluation of the proposed approach.

5.2. Decision Tree Classifier Experiment

The decision tree classifier is a hierarchically based classifier that has a tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and the leaf nodes denote the classes, an example is shown in the fig. 5.1.

For elaborating the decision trees the C5.0/See5 program [QUINLAN05] was used, that is a widely used decision tree algorithm.

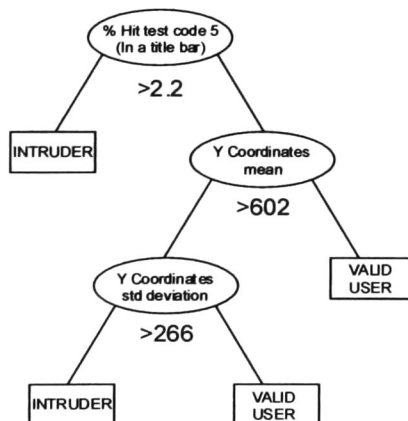


Figure 5.1. Decision tree for user #9

5.2.1. Algorithm description

The C5.0 algorithm utilizes a metric called information gain ratio that measures the reduction of the entropy in the data produced by a split; each split is determined by maximizing the reduction in the entropy of each descendant node. This course of action continues until each leaf node doesn't have gain in information or the leaf node only have elements of one class. To avoid the large decision trees that over fits the training data, the tree must be pruned back, this process is called confidence-based pruning. For more details see [QUINLAN93].

5.2.2. Experiment design

With the purpose of identifying how much time or how many mouse points are required to re-authenticate users, several experiments were implemented with distinct window sizes, dividing the 20,000 mouse data points in sets of 100, 200, 400, 500, 800, 1000, 1250 and 2000 data points to build the feature vectors; we didn't consider values larger than 2000 for the reason that it would take much time to re-authenticate the users, allowing intruders to get their objectives. Afterwards, we separated them in two groups: the training sets and testing sets; as a supervised learning approach is used, there are groups of training sets conformed by data from the valid user and data from the other remaining users labelled as intruders; this process is repeated with the remaining half with the intention of testing the decision tree classifier. In the case of the valid user sets, they are conformed by sessions of different times (beginning, middle and the ending) of the user interaction with the purpose of building an accurate user behaviour, because when the mouse data was collected, certain fatigue in the users was detected, having as a result modifications in their behaviours.

5.2.3. Results

As it can be observed in the figure 5.2 most of the users have similar anomaly detection rates with the exception of users #7, #13 and #14; this could be due to the necessity of more data points to re-authenticate them; on the other hand user #12 should have a particular characteristic that differs from the remaining users. With this empirical evaluation of the mouse-based behavioural biometric we confirm that it is possible to authenticate the users based on their mouse interaction with the system and the feasibility of the 81 features which conform the user behaviour.

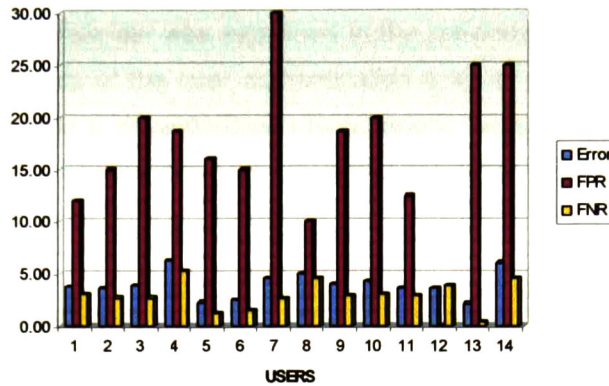


Figure 5.2. Decision tree error and anomaly detection rates with DTC

One of the objectives of this experiment is to determine what would be the minimum time to re-authenticate users, the table 5.1. shows how many points were necessary to re-authenticate users with the minimum error and it also shows the time that take to the users to generate the mouse data points; this time was taken with the assumption that when users start using the pc make more mouse movements, as an intruder would do; therefore we took the beginning time of the data gathering of each user, even though this time could vary in different applications or situations.

User	1	2	3	4	5	6	7	8	9	10	11	12	13	14	AVG
Window size	800	1000	1000	1250	800	1000	400	2000	1250	1000	1250	2000	1000	1000	1125
Time(min)	1.1	6.0	5.2	6.5	3.5	3.8	1.6	5.1	4.5	2.7	3.0	5.3	2.7	3.0	3.9

Table 5.1. User’s parameters for decision tree classifier experiment

In order to optimize the time of the re-authentication the window size should be personalized for each user, having as a consequence a different time to re-authenticate them, the average time of the authentication is 3.9 minutes as is shown in the table 5.1.

The anomaly detection rates obtained in the experiment were the following: False Positive Rate (FPR) of 19.10% and False Negative Rate (FNR) of 2.78%. The FPR is unacceptable high, however it could be reduced using another classifier; concerning to the FNR it is considered acceptable.

5.3. One-Class Statistic Classifier Experiment

The one-class statistic classifier was explained in the previous chapter, this classifier fits better to the requirements of the user authentication system, it has a low computational cost and it also provides a decentralized management, being this two advantages the reasons of its selection, besides the personalized construction and evaluation of the user behaviour.

5.3.1. Algorithm description

The algorithm has three phases: the training phase where the user behaviour is modelled; the evaluation phase where new cases are tested to see whether a valid user or an intruder is, and the third phase that consists in the followed process to automatically set the threshold for the evaluation phase (for more details see chapter 4).

5.3.2. Experiment design

It was followed a similar design than the decision tree classifier experiment, diving the 20,000 data points in sets of 100, 200, 400, 500, 800, 1000, 1250 and 2000 to build the feature vectors, afterwards the vectors were separated in two groups: the training sets and testing sets; each group is conformed by feature vectors from the beginning, middle and the ending of the session, in order to capture an accurate user behaviour. The intruder testing cases are conformed by data from the remaining users. As it is used a one class classifier, the training sets only consist of feature vectors of the target user, then the number of training vectors is set in 10 for 100, 200, 400, 500, 800, 1000 mouse data points, 8 and 5 training vectors for 1250 and 2000 mouse data points respectively.

5.3.3. Results

The figure 5.3 shows the error and the anomaly detection rates obtained per user, here we can notice the existence of two exceptional cases, the first case is the user #11 that has the largest false negative rate and the second case is the user #13 with a perfect classification of all the testing cases; we consider the existence of these exceptional cases as completely normal, since there are users which have several distinguish characteristics in their behaviour, and others which not; all the remaining users have similar values in their false acceptance/rejection rates.

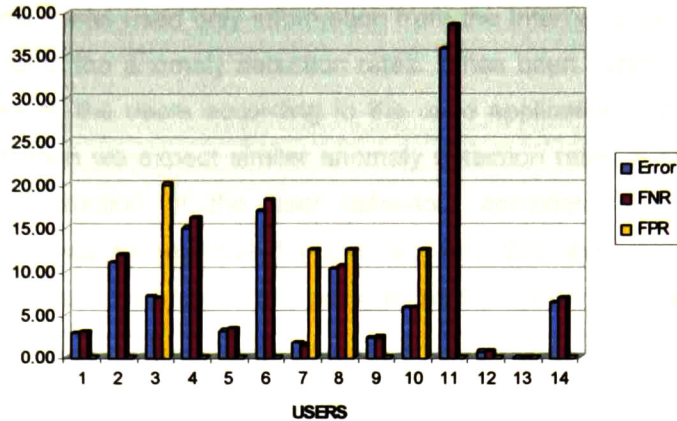


Figure 5.3. Classification errors and anomaly detection rates with OCSC

The table 5.2 shows the parameters obtained for each user in the experiment, here we can notice an increment in the required time to authenticate the users, and this could be due to the nature of the one-class approach that is only trained with feature vectors of the target user.

User	1	2	3	4	5	6	7	8	9	10	11	12	13	14	AVG
Window size	2000	1000	2000	2000	1250	1250	1000	800	1250	1250	1250	2000	2000	2000	1503.57
Threshold	917	697	723	903	639	594	876	609	528	522	558	985	1125	848	751.71
Time(min)	3.6	6.0	7.8	7.4	5.9	4.8	5.1	1.4	4.5	3.2	3.0	5.3	6.7	6.5	5.08

Table 5.2. User's parameters for one-class statistic classifier experiment

The obtained global anomaly detection rates were **FPR 2.86%** and **FNR 10.13%**; since a statistical method is used, a larger window size will result in a smaller error of classification, but it would be useless for intrusion detection; it is important to notice that the threshold was automatically adjusted, therefore the obtained behaviour of the users is more accurate than the anomaly detection rates reflect.

5.4. Conclusions

With this empirical evaluation is confirmed that is possible to authenticate the users based on their mouse interaction with the system and the usage of the GUIs, and as consequence the feasibility of the 81 features as well as the use of the OCSC as main component of the evaluation unit are proved.

In both experiments it was used only information from the Internet Explorer application to obtain clean results of the anomaly detection rates. It has been explained the presented behaviour changes by the users according to the used application and the effort of the activity performed; then we expect similar anomaly detection rates in other applications, and since the construction of the user behaviour considers the use of different applications, this will help to decrease the anomaly detection rates given that the use of each application is different for each user, giving more distinguish characteristics.

Chapter 6 Conclusions and Future Work

Summary

This chapter presents a comparison of the work presented in this thesis and some close related works, and it also discusses the conclusions and the future work

6.1. Related work comparison

It has been already mentioned the related works, however the two more related to our work are [PUSARA04] and [AHMED05]. These two works will be explained next, with the objective of comparing their results with those presented in this master thesis.

In [PUSARA04] the user behaviour is constructed using a 111 dimensional feature vector; they also used a smoothing filter overlapping the frequency of the mouse data points having as a result a high computational cost; another observation is the use of a supervised learning method, that centralizes the training phase and requires more data; the anomaly detection rates reached by them were FPR 27.5% and FNR 3.06% with an average time of 4.48 minutes. In our work we use only 81 features to construct the behaviour of the user, reducing the computational cost, without using a smoothing filter, and the addition of the one class classification gives many advantages to our work such as decentralized management, fast training and less data collection; besides the anomaly detection rates reached in our experiments were better with a similar average time of the re-authentication.

In the work described in [AHMED05] a mixture of the mouse dynamics and keystroke dynamics is proposed, obtaining the following results FPR 1.31% and FNR 0.65%; they use a key oriented neural network based approach to simulate keystroke dynamics with reference to other keys to approximate a digraph/trigraph value based on other detected graphs of the users and the locations of the keys with reference to each other; on the side of the mouse dynamics they used a statistical method similar to ours, nonetheless the required amount of data to enroll the users in their system is very large, increasing the required time in the enrollment phase and the computational cost in the training phase of the neural network; having as a result an impractical registration of new users in the system, comparing it to our method using the OCSC that only needs 10 feature vectors to construct the user behaviour for one application, and with very low computational cost, taking into account only information coming from the mouse device.

None of the presented related works took into consideration the changes in the user behaviour due to the adaptation to the systems, changes in the hardware, small changes in the GUIs, etc., then the duration of the validity of the user behaviour will be affected

increasing the false alarms, contrary to our work that make use of a feedback loop that can absorb these variations.

6.2. Conclusions

It has been shown in a empirical way, the capability of authenticating users utilizing the mouse based behavioural biometric and the usage of the GUIs; by building the behaviour of the users we can transparently authenticate them, with a free interaction with the computer system, using an anomaly detection approach; however our approach don't cover all the masquerading scenarios such as when the user avoids the use of the mouse device; therefore with our study is pretended to complement the keystroke dynamics approach, that is also insufficient given the increasing usage of the graphical user interfaces where mouse devices are commonly used.

In this work several improvements to related works are presented, such as: the reduction of the number of features considered to construct the user behaviour and the reduction of the required data in the enrollment phase of the users, resulting in a lower computational cost in the enrollment and the verification phases; in addition we have identified new features extracted from the user interaction via the computer mouse such as the hit test codes and the click duration.

Regarding the application approach, we explained the behaviour changes presented by the users according to the application and the effort of the activity performed, being a fundamental part in the construction of the user behaviour that helps to reduce the time to authenticate the users.

By building the user behaviour with mouse information, the use of expensive hardware is avoided, and in certain way the mouse data is easily obtained through the user interaction with the pc, moreover we are assured that mouse-based behaviour of the user can't be imitated by other users.

6.3. Future work

The direction of our future work will be directed to complement our re-authentication method with the keystroke dynamics to provide a complete solution for user authentication, once delimiting the responsibility to the current user, we would like to combine other kind

of intrusion detection systems based on systems calls to detect a wide range of internal attacks, and also the combination of network traffic monitoring should be necessary to create a robust intrusion detection system.

References

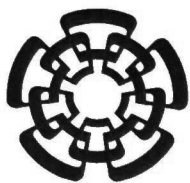
- [AHMED05] A. A. E. Ahmed, and I. Traore. Detecting computer intrusions using behavioral biometrics. In the third annual conference on privacy, security and trust, October, 2005.
- [BIOPASSWORD06] BioPassword
<http://www.biopassword.com/>.
- [BRO06] Bro Intrusión Detection System
<http://bro-ids.org>
- [BURGESS01] Mark Burgess, Harek Haugerud, Trond Reitan, and Sigmund Straumsnes. Measuring system normality. Association for computing machinery, Inc., Classification, USA 2001.
- [CANAVAN00] J. E. Canavan. Fundamentals of Network Security. Artech House 1st Edition, 2000.
- [CISCO06] Cisco Secure Intrusion Detection System
<http://www.cisco.com/univercd/cc/td/doc/product/iaabu/csids/csids1/csidsug/overview.htm>
- [CUNNINGHAM01] E. P. Cunningham and C. M. Meghen. Biological identification systems: genetic markers. Rev. sci. tech. Off. Int. Epiz., 20 (2), pp. 491-499, 2001.
- [DAUGMAN04] J. Daugman. How Iris Recognition Works. IEEE Trans. on Circuits and Systems for Video Technology, Vol. 14, No. 1, pp. 21-30, January 2004
- [DELAC04] Kresimir Delac and Mislav Grgic. A survey of Biometric Recognition Methods. In the 46th International Symposium Electronics in Marine, ELMAR-2004, Zadar, Croatia, June 2004.
- [DENNING87] D. E. Denning. An intrusion-detection model. IEEE Transactions on software engineering, Vol. SE-13, No. 2, February 1987, pp. 222-232.
- [DITYAN03] Y. Dit-Yan and Y. Ding. Host-based intrusion detection using dynamic and static behavioral models. Pattern Recognition, Volume 36, Issue 1, January 2003, pp. 229-243.
- [ENZHE04] Enzhe Yu, Sungzoon Cho. Keystroke dynamics identity verification-its problems and practical solutions. Computer and Security, Vol. 23 Issue 5, July 2004, pp. 428-440

- [GAINES80] Gaines R., Lisowsky W., Press S., and Shapiro N. Authentication by keystrokes timing: some preliminary results. Rand Report R-256-NSF. Rand Corporation; 1980.
- [GAMBOA03] H. Gamboa and A. Fred. An identity authentication system based on human computer interaction behavior. In Jean-Marc Ogier and Eric Trupin, editors, *Pattern Recognition in Information Systems – Proc. of the 3rd Intl. Workshop on Pattern Recognition in Information Systems*, pages 46 –55. ICEIS PRESS, 2003.
- [GAMBOA04] H. Gamboa, A. Fred. A Behavioral Biometric System Based on Human Computer Interaction. In *Proc. of Proceedings of SPIE, Vol. #5404-36*, 2004.
- [GHOSH99] A. K. Ghosh, A. Schwartzbard, and M. Schatz. Learning Program Behavior Profiles for Intrusion Detection. In *Proceedings 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, Santa Clara, California, April 1999, pp. 51-62.
- [GOLDRING03] T. Goldring. User profiling for intrusion detection in windows NT. In *Proceedings of the 35th Symposium on the Interface, Computing Science and Statistics Volume 35 Security and Infrastructure Protection*, Salt Lake City, Utah, March 2003
- [GRANDVAUX04] O. Grandvaux. How to design security to prevent internal security threats. Department of Computer and System Sciences, Stockholm University / Royal Institute of Technology, 2004.
- [GUPTA97] J. Gupta and A. McCabe. A review of dynamic handwritten signature verification. Tech. rep., James Cook University, Australia, <http://citeseer.nj.nec.com/gupta97review.html>, 1997.
- [HAKAN00] A. Hakan. Network and agent based Intrusion Detection Systems. TU Munich, Department of Computer Science, 2000.
- [HANSHENG05] Hansheng Lei, S. Palla, V. Govindaraju. Mouse-based signature verification for secure Internet transactions. The 17th SPIE conference, *Applications of Neural Networks and Machine Learning in Image Processing IX*, California, USA, January, 2005
- [HASHIA05] S. Hashia, C. Pollett, M. Stamp. On using mouse movements as a biometric. Int Proceedings of ICCSA 2005, P.P. Dey and M. N. Amin, editors, San Diego, California, June, 2005.
- [HOCKET04] Hocket Sylvain, Ramel Jean-Yves, Cardot Hubert. User Authentication By a Study of Human Computer Interactions. The 8th Annual Meeting on Health, Science and Technology, France, 2004.

- [JAIN04] Anil K. Jain, Arun Ross and Salil Prabhakar. An introduction to biometric recognition. In the IEEE Transactions on circuits and systems for video technology, vol. 14, No. 1, January 2004.
- [JOO06] Hyoung-joo Lee and Sungzoon Cho. Retraining a novelty detector with impostor patterns for keystroke dynamics-based authentication. IAPR International Conference on Biometrics, Hong Kong, 5-7 January 2006.
- [KALEZHI04] Kalezhi Josephat. Mouse Biometrics. Thesis of MSc Information System, School of Computing, University of Leeds, United Kingdom, 2004.
- [KAMEL04] M. Kamel, C. Zhang, and J. Ju. Intrusion detection using hierarchical neural networks. 2004.
- [KOROTKAYA] Z. Korotkaya. Biometric Person Authentication: Odor. Advanced 1topics in information processing, 2003.
- [KUMAR04] Kumar S. Hybrid Profiling Strategy for Intrusion Detection. Department of Computer Science – University of British Columbia, 2004.
- [LANE97] T. Lane and C. E. Brodley. An Application of Machine Learning to Anomaly Detection. Proceedings 20th {NIST}-{NCSC} National Information Systems Security Conference, 1997, pp. 366-380.
- [MAIO02] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, and A. K. Jain. FVC2002: Fingerprint verification competition. In Proc. Int. Conf. Pattern Recognition (ICPR), Quebec City, QC, Canada, Aug. 2002, pp. 744–747.
- [MARCHETTE99] David J. Marchette. A statistical method for profiling network traffic. In Proceedings 1st USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, California, April 1999, pp. 119-128.
- [MARIN01] J. Marin and D. Ragsdale and J. Surdu. A hybrid approach to the profile creation and intrusion detection. DARPA Information Survivability Conference and Exposition (DISCEX II'01) 1, 2001.
- [MAXION04] Roy A Maxion, and Tahlia N. Townsend. Masquerade detection Augmented with error analysis. In the proceedings of IEEE Transactions on reliability, Vol 53, No. 1, March 2004.
- [MONROSE00] Monrose, F., Rubin, A.D.: Keystroke Dynamics as a Biometric for Authentication. Future Generation Computer System 16(4) (2000) 351-359.

- [NIXON04] Nixon, M.S. Carter, J.N. Advances in automatic gait recognition. Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004 pp. 139-144.
- [NORTON06] Symantec Worldwide
<http://www.symantec.com>
- [PHILIPS02] P. J. Philips, P. Grother, R. J. Micheals, D. M. Blackburn, E. Tabassi, and J. M. Bone. FRVT 2002: Overview and Summary
<http://www.frvt.org/FRVT2002/documents.htm>
- [PUSARA04] M. Pusara, C. E. Brodley, User re-authentication via mouse movements. In Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security, Washington DC, USA, 2004.
- [QUINLAN93] J. R. Quinlan. C4.5: Programs for machine learning. Morgan Kaufmann, San Mateo, CA, 1993.
- [QUINLAN05] J. R. Quinlan. Data mining tools See5 and C5.0
www.rulequest.com/see5-info.html. 2005
- [RIVEST78] R. Rivest, A. Shamir, L. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM, Vol. 21 (2), 1978, pp. 120-126.
- [ROSS03] Ross A. J. Everitt and Peter W. McOwan. Java-Based Internet Biometric Authentication System. In the IEEE Transactions on Pattern analysis and machine intelligence, Vol. 25, No. 9, September 2003.
- [SCHNEIER04] B Schneier. Sensible Authentication. Queue Vol. 1, No. 10, 2004, pp. 74-78.
- [SCHONLAU01] Matthias Schonlau, William DuMouchel, Wen-Hua Ju, Alan F. Karr, Martin Theus and Yehuda Vard. Computer Intrusion: Detecting Masquerades. Statistical Science Vol. 16, No. 1. 2001, pp. 1-17
- [SOCOLINSKY03] D. A. Socolinsky, A. Selinger, J. D. Neuheisel. Face recognition with visible and thermal infrared imagery. Computer vision and image understanding Vol 91, Issue 1-2, 2003, pp. 72-114.
- [STANIFORD96] S. Staniford-Chen and S. Cheung and R. Crawford and M. Dilger and J. Frank and J. Hoagland and K. Levitt and C. Wee and R. Yip and D. Zerkle. GrIDS - A Graph-based Intrusion Detection System for Large Networks. In Proceedings of the 19th National Information Systems Security Conference. 1996.
- [TMF06] The Three Myths of Firewalls
<http://web.mit.edu/kerberos/www/firewalls.html>

- [WONG03] David C. M. Wong, Jay Kumar, Helen C. Shen, Anil K. Jain. Personal verification using palmprint and hand geometry biometric. Audio- and Video-Based Biometric Person Authentication: 4th International Conference, AVBPA 2003 Guildford, UK, June 9-11, 2003 Proceedings. ISBN: 3-540-40302-7. pp. 668 - 678
- [YE01] Nong Ye, Xiangyang Li, Qiang Chen, Syed Masum Emran, and Mingming Xu. Probabilistic Techniques for Intrusion Detection Based on Computer Audit Data. In Proceeding of IEEE Transactions on systems, man, and cybernetics – part A: Systems and humans, Vol. 31, No. 4, July 2001.



CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N. UNIDAD GUADALAJARA

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional aprobó la tesis

Autenticación de Usuarios usando Biométricas del Ratón

del (la) C.

José Octavio GUTIÉRREZ GARCÍA

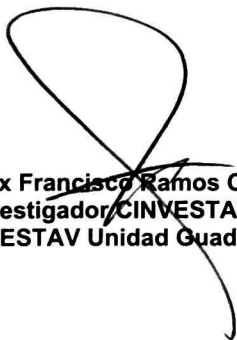
el día 8 de Septiembre de 2006.



**Dr. Juan Manuel Ramírez Arredondo
Investigador CINVESTAV 3B
CINVESTAV Unidad Guadalajara**



**Dr. Luis Ernesto López Mellado
Investigador CINVESTAV 3A
CINVESTAV Unidad Guadalajara**



**Dr. Félix Francisco Ramos Corchado
Investigador CINVESTAV 2B
CINVESTAV Unidad Guadalajara**



**Dr. Mario Angel Siller González Pico
Investigador CINVESTAV 2A
CINVESTAV Unidad Guadalajara**



CINVESTAV
BIBLIOTECA CENTRAL



SSIT000008744