Journal of Computer Science 10 (11): 2299-2314, 2014
ISSN: 1549-3636

doi:10.3844/jcssp.2014.2299.2314 Published Online 10 (11) 2014 (http://www.thescipub.com/jcs.toc)

DESIGN AND IMPLEMENTATION OF THE TPM USER AUTHENTICATION MODEL

Marwan Ibrahim Alshar'e, Rossilawati Sulaiman, Mohd Rosmadi Mokhtar and Abdullah MohdZin

Research Center for Software Technology and Management (Softam),
Faculty of Information Science and Technology, National University of Malaysia, Malaysia

Received 2014-06-06; Revised 2014-07-23; Accepted 2014-11-28

Competing Interests: The authors have declared that no competing interests exist

ABSTRACT

The Trusted Computing Group (TCG) has introduced the Trusted Platform Module (TPM) as a solution to assure end users of their privacy and confidentiality. Although the TPM is designed to prevent software attacks, the TPM itself is vulnerable to physical attacks that could enable intruders to gain access to confidential data. In general, the TPM provides an ID and implements a password identification technique to prevent unauthorized users from gaining access to the TPM. The TPM user authentication is carried out by the TPM itself, which exposes the TPM to direct risk as highly skilled intruders can break the authentication line of defence and gain access to the TPM. The process of encrypting and decrypting information, especially when asymmetric algorithms are used, is viewed as a process that consumes time and resources, which decreases the speed of the computer. In order to solve the problems, a TPM User Authentication Model (TPM-UAM) that can provide the TPM with a higher level of security and resistance against physical attacks has been proposed as we proposed in our previous research paper (Alshar'e *et al*., 2014). The technique is based on biometric authentication to prove the identity of the users and to allow the process of authentication to happen at an independent platform using virtualization that will keep the TPM out of reach until a user is completely verified and approved. The TPM-UAM is able to provide a more satisfactory level of confidence for data and processes that can be rated as highly confidential and private. The model was successfully developed and tested and the results confirmed the model efficiency and ability to secure TPM and all functions have been confirmed to be working perfectly according to what they were designed for. This paper describes the design and implementation of TPM-UAM system based on the proposed authentication model, virtualization has been implemented to create authentication platform to prevent direct interaction with TPM and biometrics has been implemented to verify identities and supervise running TPM, the system testing results in confirming the system functionality and ability to secure and protect TPM.

Keywords: TCG, TPM, Authentication, Biometrics, Face Recognition, Fingerprint, Virtualization, Xen

1. INTRODUCTION

The Trusted Computing Platform (TPM) was developed by the Trusted Computing Group (TCG) to include additional hardware and software to increase the security level of information systems. The current and common use and implementation of TPM is a small chip placed at the main board which can store cryptographic

Corresponding Author: Rossilawati Sulaiman, Research Center for Software Technology and Management (Softam), Faculty of Information Science and Technology, National University of Malaysia, Malaysia

keys and other critical security information and provide cryptographic functions like asymmetric encryptions and signature schemes (TCG, 2010).

Even though the TPM is designed to prevent attacks to the software, the TCG claims that not enough effort was spent on avoiding physical attacks to the TPM (TCG, 2010) as the main concentration was on the implementation of PIN password authentication methods to confirm user identity. Klenk *et al*. (2009) reported that the TPM authentication alone is not a significant solution to confirm and verify users' identities.

The common practice in a computer system with confidential information in it, user relies on the encryption techniques to protect their confidential information against various risks and attacks such as in the case of physical attacks when computer system or hard drive is stolen, where the thief cannot read the confidential information without the proper decryption keys (Müller *et al*., 2013). Rutkowska and Tereshkin (2009) describes the Evil Maid attack mechanism as a major risk and attacking mechanism whereby an attacker can make his way to access protected personal computer systems and collect confidential information. To prove the idea of the Evil Maid they implemented an attack against a computer system with full disk encryption using the True Crypt software system. In addition user also might keep unattended computers in a switched off state to ensure security, according Rutkowska and Tereshkin (2009) this is not a problem for the attacker considering the Evil Maid scenario, when a third party user (attacker) obtain physical access to personal computer system, attacker can boot the system from Evil Maid USB stick, then to leave the machine until the authorised user returned and logged in to the system. At this time Evil Maid collects passphrases and then stores it in the hard drive, or even transmits it over the network to be used later by the attacker. After the attacker received the passphrases he/she needs to get access or even steal the computer system with the collected passphrases from Evil Maid he/she can read and get encrypted information. Evil Maid attack still a threat against True Crypt as well as Bitlocker (Müller *et al*., 2013; Götzfried and Müller, 2014).

Implementing methods and techniques to verify and authenticate users before they are able to use the TPM will increase the security level and protection of the TPM. This is possible by ensuring the identity of the user before loading the TPM functions onto the registers and opening it to other software.

In order to solve the problem, we proposed a user protection Model is proposed for the trusted computing environment (Alshar'e *et al*., 2014) and it has been demonstrated that the TPM user authentication model is a solution to overcome weaknesses within the TPM and to provide a solid basis for authenticating and confirming the identity of the user, thus protecting the TPM from unwanted access and keeping it safe against Evil Maid and other similar attack mechanisms until verification of the user has been completed.

The TPM-UAM provides a security guard for the TPM, preventing unauthorized users from interacting directly with the TPM by providing a second platform on the same machine to handle the TPM, while the main platform performs the user authentication separately. Once the authentication platform is on, then the TPM-UAM directs the user into a number of identity verification processes, as seen in **Fig. 1**.

As proposed in our model at Alshar'e *et al*. (2014) The TPM User Authentication Model consists of three main stages, which are, firstly, prevent direct interaction between unauthorised users and TPM, as the model proposed this prevention occurs through virtualization concept, where two platforms will be created using virtualization, the first platform will handle users authentication including unauthorised users and the second platform will run the TPM, this prevention shall protect TPM against attacks such as in Evil Maid as proposed early, where only verified identities only shall interact with TPM, secondly, the user Authentication and Verification stage, which will be responsible for verifying the user identity using biometric techniques which shall support the weak authentication mechanism in TPM and finally, ensure user privacy and protect running TPM platform (after authorised user logged in to platform with TPM) by monitoring the number of users present in front of the PC all the time. Refer to our proposed model Alshar'e *et al*. (2014) for a full description of the model.

This study, which describes the design and implementation of a system based on the proposed model, is divided into six sections. The next section describes the design considerations for the system, followed by the system design and implementation in the next two sections. Section 5 discusses the results of the experiment, while the last section is the conclusion.

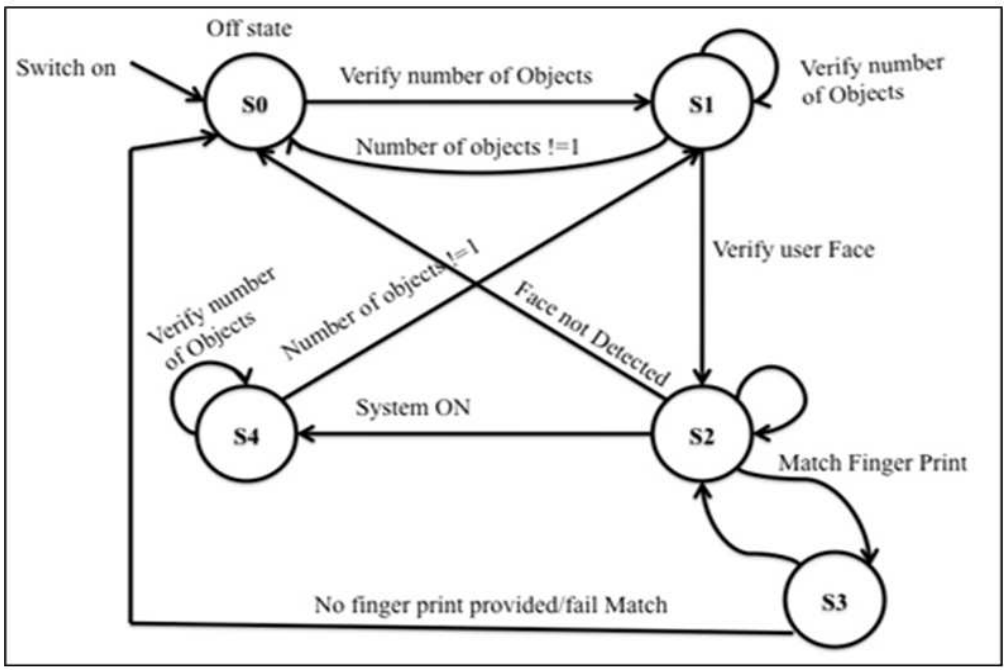

Fig. 1. TPM-User Authentication Model (Alshar'e *et al*., 2014)

2. DESIGN CONSIDERATIONS

2.1. Requirement Analysis

To provide a better understanding of the functionality of the system based on the model shown in **Fig. 1**, a list of functional requirements have been stated to guide the process of analysing the situation. Each functional requirement describes one of the functions of the system or what the system should do or provide as a service. The functional requirements are as follows:

- The system should provide the user with the ability to start a secure session on a secure platform with the TPM
- The system should confirm the presence of a single user at the time of interaction with a secure platform
- The system should authenticate the user before allowing him to interact with the secure platform
- The system should verify the user's fingerprint with the stored image of the authorized user
- The system should verify the image of the user's face with the stored image of the authorized user
- The system should monitor the number of users during any or all of the running sessions
- The system should start the VMM and run the virtual platform which contains the TPM

For a better understanding and to provide a single view of the system and the system's functionality, the UML use case is provided, which describes the main use cases and the main actors with the system, as shown in **Fig. 2**.

As shown in **Fig. 2**, the system is supposed to start working when a user tries to start a session to gain access to the platform with the TPM on it. Once the trigger is pulled, the system is supposed to start working by reading the user in front of the PC to verify the number of users in front of the PC.

Once the number is confirmed, the system will run the Manage Session function. The Manage Session will call up the Authenticate User function, which in turn will call up the Verify User function to read and confirm the user's fingerprint via the Fingerprint Detection function and the user's face via the Face Recognition function. Once the user has been verified, the Manage Session function will call up the Virtual Machine Manager (VMM) to start the work and to launch the virtual platform.

2.2. Authentication Method Selection

Biometric systems work by extracting key features of people and comparing them with the stored images of

these features within a database to verify the identity of a person. The two main purposes of biometrics are the *verification* mode (*Am I who I claim I am?*) and the *identification* mode (*Who am I?*) (Jain *et al*., 2004; Khushk and Iqbal, 2005; Zajkowska *et al*., 2007).

The performance of the different biometrics can be assessed based on various characteristics such as accuracy, speed and storage as well as the cost and ease of use (Jain *et al*., 2004). To adopt a biometric technique, several characteristics should be investigated. Jain *et al*. (2000; Prabhakar *et al*., 2003) have stated three main characteristics, namely (1) *universal* characteristic, where each individual is the only one who has this characteristic and there is no chance to find another person who has it, (2) *permanent* characteristic, which is not subject to change in the future and (3) *collectable* characteristic, where this characteristic is easily captured by a sensor and can be measured. Other issues to be considered in relation to biometric selection are (4) *performance*, that refers to the accuracy, speed, resources of the requirements and the factors or the operations that can affect the performance of the system. In addition (5) *acceptability* refers to how much are the people prepared to use or deal with these systems in their daily lives and (6) *circumvention* refers to how easy it is to break into the system using fake characteristics or methods (Jain *et al*., 2000; Prabhakar *et al*., 2003).

According to Zajkowska *et al*. (2007), biometric systems act as a key which will allow users to move to the next stage after successful identification. When a biometric system recognizes a biometric feature, an action should occur through another device or system which is connected or attached to the biometric system.

Biometric devices are used in different organizations for different purposes such as access control and work time registration. For example, when an employee requests access to a room, he/she should present the requested biometric feature to the scanner so that when verification and recognition have been completed, the system will unlock the door. **Table 1 and 2** show various comparisons that have been made concerning some of the most popular authentication methods.

This research suggests using biometric authentication methods instead of PIN/Password authentication methods. Based on the literature review of biometric selection and the nature of the desired model and in order to ease the use of biometric techniques within the model, fingerprint and face recognition techniques have been chosen for implementation within this model.

The fingerprint technique has been suggested due to its high accuracy, low error rate and high reliability. In addition, fingerprint devices and techniques are considered to be low cost techniques and furthermore, they are easy to use and do not require the users to be highly skilled. Therefore, the fingerprint technique is used here as a first attestation technique to prove the user's identity and that the user is registered.

The face recognition technique is also considered to be accurate, reliable and easy to implement at low cost. Face recognition through the iris or retina is not reliable and accurate, but in terms of user acceptance and cost and to match the nature of the model, where the user has to be monitored all the time and also since in this model face recognition is used to confirm the user based on his fingerprint (if in ID/Password, fingerprint will be the ID and face will be the Password, then the system will open only when there is a match), face recognition has been found to be the best match for this model. On the other hand, fingerprint and face recognition devices have a high impact, where these can be easily implemented and installed within any device such as mobile phones, desktops and laptops.

2.3. Virtualization Technology

Virtualization can be seen as the simultaneous execution of more than one Operating System (OS) instance on a single computer (Bower, 2010). The term 'virtualization' is widely known as the separation of resources, or in general, a service request by its underlying physical delivery. With virtual memory, for example, the computer software accesses more memories than what has been physically installed by using background swapping of data to disk storage. However, virtualization techniques can be applied to other IT infrastructural layers including networks, storage, laptops or server hardware, operating systems and applications (Dalton *et al*., 2009).

Virtualization gives meaning to the logical running of applications or OS, which are basically taken from the current available physical resources, but which function as separate devices so as to guarantee that there is no interference between the resources or the devices after the creation of the logical components and that each is run independently (Hagen, 2008). A study by Bower (2010) clarified that the use of virtualization for server rooms could increase the reliability and utilization of high end servers and will cut the operational cost of the server.

Virtualization works to partition a single physical server into many logical servers, where each logical server will be able to run an OS and its applications separately and independently (Ray and Schultz, 2009). Among the benefits of virtualization are:

- *Cost saving*, since organizations can use the same physical machine to do different tasks which, without virtualization, require separate servers to do them
- *Dynamic load balancing.* While running different application son the same machine can cause the system to slow down or even halt due to the occurrence of a processing bottleneck, a dynamic load will prevent this from happening by running the different applications continuously
- *Lower power consumption.* This refers to the usage of the servers themselves, where logical servers and cooling facilities are used on the same machine
- *VMs used to isolate processes from attackers and malware.* The user access to the applications can be easily controlled, where virtual machines can make some applications inaccessible to some users and restrict user interaction to a particular area or server and where malicious attacks can be reduced. The hypervisors split physical resources into isolated entities where each guest OS runs independently. Each OS encapsulated and abstracted from the hardware. The abstraction of physical resources provides additional security as each VM allocates its own resources and bounds it to itself. The OS controls hardware access by extracting the hardware details and then to communicate directly with the hardware. VM works on segment of hardware resources, which would make it possible to monitor the use of VM to the allocated resources and detect unwanted behaviour by any malicious software

2.3.1. Xen Virtualization

The University of Cambridge launched the Xeno Servers project as a new technology to develop a powerful and flexible infrastructure which will help to enable single machines to run different or multiple operating systems and applications in isolated and secure environments (Hagen, 2008). XEN is an open-source para-virtualization technology that provides a platform for running multiple operating systems on one physical hardware resource.

XEN supports several operating systems, e.g. Linux, FreeBSD, Windows and Net BSD. It enables users to easily test, deploy and run their software and services on multiple operating systems with resource isolation and great performance (Chaganti, 2007).

According to Abels *et al*. (2005), "*Xen virtualization technology-available for the Linux kernel-is designed to consolidate multiple operating systems to run on a single server, normalize hardware accessed by the operating systems, isolate misbehaving applications and migrate running OS instances from one physical server to another*". In other words, the Xen software layer, which lies directly on the top of the hardware, takes the place of the operating system to allow the computer to run multiple operating systems at the same time.

Xen runs its hypervisor on ring 0 to control and monitor the access to the hardware between the virtual machines. It controls and manages the access to the memory and the hardware through the Xen hypervisor and the privileged Xen-modified kernel (Hagen, 2008).

2.3.2. Protection and Isolation of Computer Processors

According to Hagen (2008), processors typically run in different protection modes on purpose to stop and capture unauthorized access to the physical processors and device resources. The X86 runs three protection modes, which are the real mode, System Management Mode (SMM) and protected mode. At the protected mode, it provides multiple protection levels which are needed to differentiate between the required type of access to the memory and the general hardware by the operating system and by higher-level applications. The protected mode has four levels of protection, called rings, which are numbered from 0 to 3.

In a normal operating system environment, the operating system kernel runs in ring 0. Thus, it can control the execution of all privileged instructions, manage the memory and directly specify the range of memory addresses available to an operating system. Thus ring 0 is known as the "supervisor mode" or "kernel mode" because it directly controls the hardware and memory access (Gareau, 1998).

The code which is running on ring 0 will decide the protection level of each code segment since the OS instructions will be loaded onto the ring. This means that the instructions or code loaded by the OS has a higher-level privilege to load and execute other code segments. Different privileges for any code indicate whether the system will load this instruction or not and to which protection level it will be sent. This approach in monitoring the different processes is known as the

supervisor mode and is supported by the Operating System. A common problem related to this approach is that some privileged instructions fail to trap when they are executed with insufficient privileges, thus making it possible for them to be executed at the wrong modes or unknown states.

2.3.3. Xen and Other VMs

Xen hypervisor has been chosen over other virtual machine hypervisors such as VMware ESXi, Hyper-V and KVMe, for a number of reasons. Firstly, the guest OS requires para-virtualized drivers and can be seen as a thin hypervisor model that can run directly on the hardware. Secondly, Xen allows the guest operating system to co-operate with the hypervisor to improve the overall performance of the I/O, CPU and memory virtualization. Thirdly, Xen relies on service domains for its functionality. Fourthly, XEN separates the hypervisor execution from the management of the OS, the management stack, device drivers and guests (components). Fifthly, there is a strong isolation between all the components. Sixthly, the domains can restart without taking out the full system. Seventhly, Xen is very scalable and finally, Xen ensures a high level of security and reliability (Spector and Xen, 2011).

2.3.4. Xen Hypervisor

The Xen hypervisor requires a modified version of the operating system for the virtual machine when the hardware does not support virtualization and these modifications will include:

- Avoidance of privileged instructions by using an abstract hardware model that differs from the specific hardware that is available on the physical machine
- Execution at a lower privilege level than the hypervisor

The hypervisor will be responsible for monitoring and controlling the memory management, CPU exception handlers, system calls, hardware interruptions, timers and all direct I/O devices. This virtualization approach is called para-virtualization since modifications are required for the OS to handle privileged operations by communicating with the hypervisor or Xen-modified kernel (Hagen, 2008).

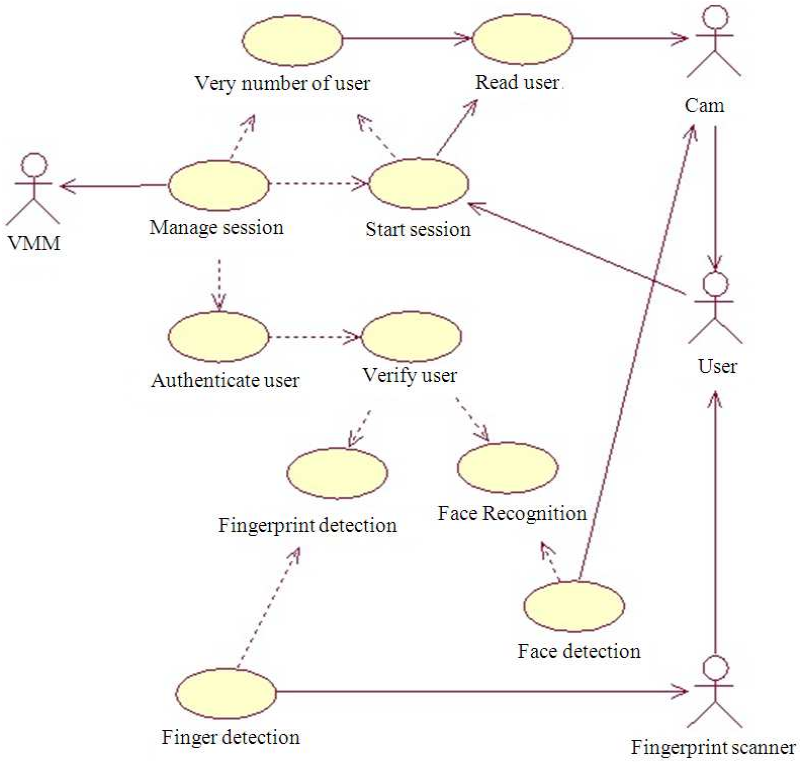

Fig. 2. TPM-UAM use case diagram

Table 1. Comparison of biometric technologies (Zajkowska *et al*., 2007)

Recognition Method	Usefulness at verification	Usefulness at identification	Accuracy	Reliability	Error rate
Fingerprints	✓	✓	••••	•••	1:500
Face	✓	×	•••	••	lack of reliable data
Palm geometry	✓	×	•••	••	1:500
Speaker's voice	✓	×	••	•	1:50
Iris	✓	✓	••••	•••	1: 131000
Retina	✓	✓	••••	•••	1:10000000

Table 2. Comparison of biometric techniques (Liu and Silverman, 2001; Khairwa and Anshul, 2012)

Recognition method	Most common reasons for error	Security level	Ease of use	Low Implementation costs	Equipment type and cost
Fingerprints	skin moisture, dirt, identified person's age	•••	•••	✓	specialist, cheap
Face	bad light, age, glasses, moustache/beard	••	••	✓	simple, cheap
Palm geometry	Injuries, age	••	•••	×	specialist, medium
Speaker's voice	noise in the background	••	•••	✓	simple, cheap
Iris	Bad light	•••	••	×	specialist, medium
Retina	Glasses	•••	•	×	specialist, expensive

3. OVERALL SYSTEM DESIGN

The TPM User Authentication Model consists of two main parts. The first part comprises the setting up of a Xen hypervisor for the preparation of two platforms. The main platform is hosted by a Xen hypervisor to help create a secure platform and to perform user authentication and verification, while the second platform will handle the TPM and can only be reached via the main platform and through the authentication and verification processes.

The second part comprises the user authentication and verification processes, as discussed in section 2.2 Biometric authentication, which consists of face recognition and fingerprint authentication, is deployed as a two-way verification technique to authenticate users instead of a PIN/password. Following that, the number of users' detection is implemented to detect the number of users in front of the PC in order to protect user privacy. This is performed by counting the number of users present in front of the screen. If the number of users is zero or more than one, the platform will suspend the screen, making it blank. **Figure 3** shows the structure and subsystems of the verification processes.

3.1. System Structure

A fingerprint scanner reads an image of a user's fingerprint and sends it to a feature extractor to collect the required features for recognition. It then goes through a finger recognition process, where the database is searched for a match for the taken image.

At any time when there is only one user in front of the cam, the system will detect the face of the user and then pass it to the feature extractor to prepare the mage for the recognition process; the face recognition will search the database for a match to the extracted face image.

When the user is identified as a registered user, it means that the fingerprint has been found in the database. Then the detected face image is matched with the face image associated with that fingerprint before a decision is taken to open the session for the user.

3.2. System Activity Diagram

To implement the desired system, it is suggested that the system work and act according to the activity diagram as presented in **Fig. 4**.

When the user wants to start the secure platform which has the TPM control, the system will start to work. As shown in **Fig. 4**, the system acquires the number of users in front of the PC. As the model suggests, in order to protect the user's privacy, the system will ensure that there is only one user interacting with computer and it will not make any further moves until it has confirmed that there is only one user in front the PC.

The system will confirm if it is a first run or the returning process. If it is a first run, the VM will still be closed, but if it is a returning process that means the VM will be in the ON state and a trigger is pulled. Then the system can go back to the process of acquiring the

number of users, which will determine its next step. If it is a returning process, the system will need to confirm the face and continue to open the VM again. If the face recognition fails, then the system will need to read the fingerprint to confirm the identity.

In the case where the VM is closed, this indicates that it is the first run of the system. As such, the system will ask for the user's fingerprint to confirm the user's identity (to prove that the user is a registered user). When the system finds a match, it will proceed to verify the user's face by face recognition. The image that is taken of the face is supposed to match the face image that is associated with the fingerprint image. When a match is found, the system will open and start the VM.

If a match is not found, the system will go back to confirm the user's identity (fingerprint), then read the face again to find a match. This process will be repeated three times. If the system fails to find a match the third time, then it will close without opening the VM.

When the system and the VM are ON, the system will continue to verify the number of users all the time in order to detect any other person who comes in front of the PC or to detect the absence of the authorized user. If the number of users at any time is not equal to one, the system will display a screensaver to block the view to the whole system and the user, as the owner of the PC, will have to provide a password to unlock the screensaver. The system will then proceed to verify the user's authority and identity before returning to the process of acquiring the number of users. The flow will then resume.

3.3. System Design

Figure 5 shows the diagram of the package system structure, where the system is supposed to contain six main packages which will work together in such a way as to bring the system to life. The *UI-auth-sys* is the interfacing package which will handle the interfaces to be displayed for the user and the user's interaction with the system. The Face recognition package contains the *qt-authsys* and *App Stataus* classes, which will handle the face detection and recognition processes and the other parts of the system, as explained in the class diagram description.

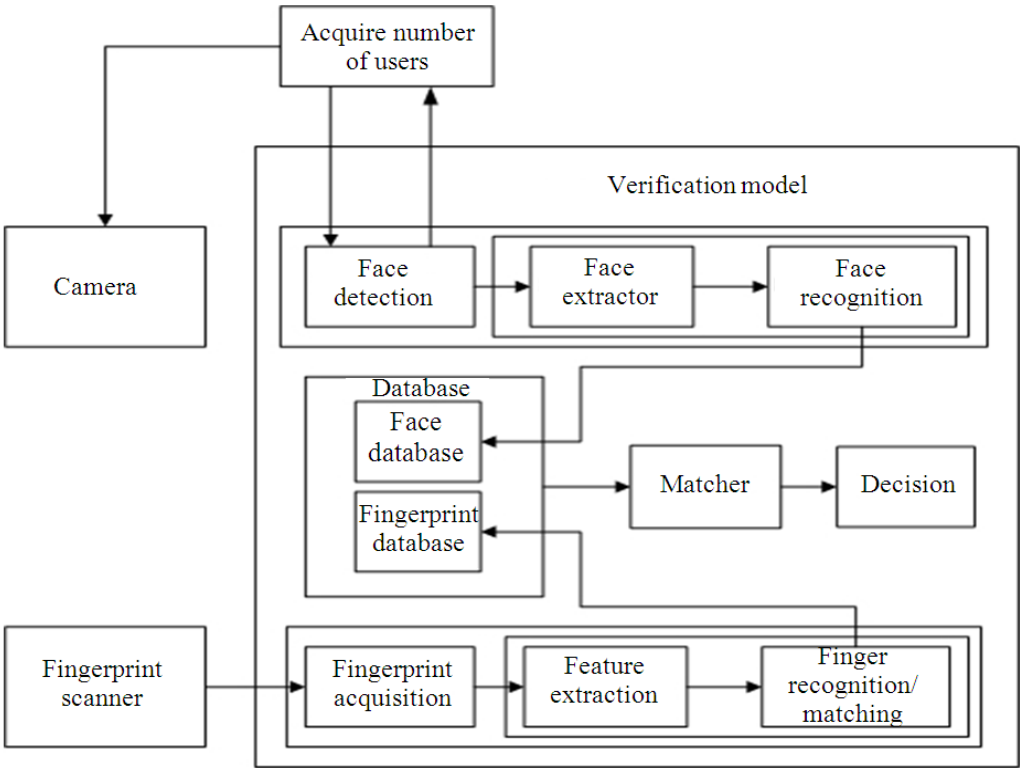

Fig. 3. Block diagram for user authentication and verification processes

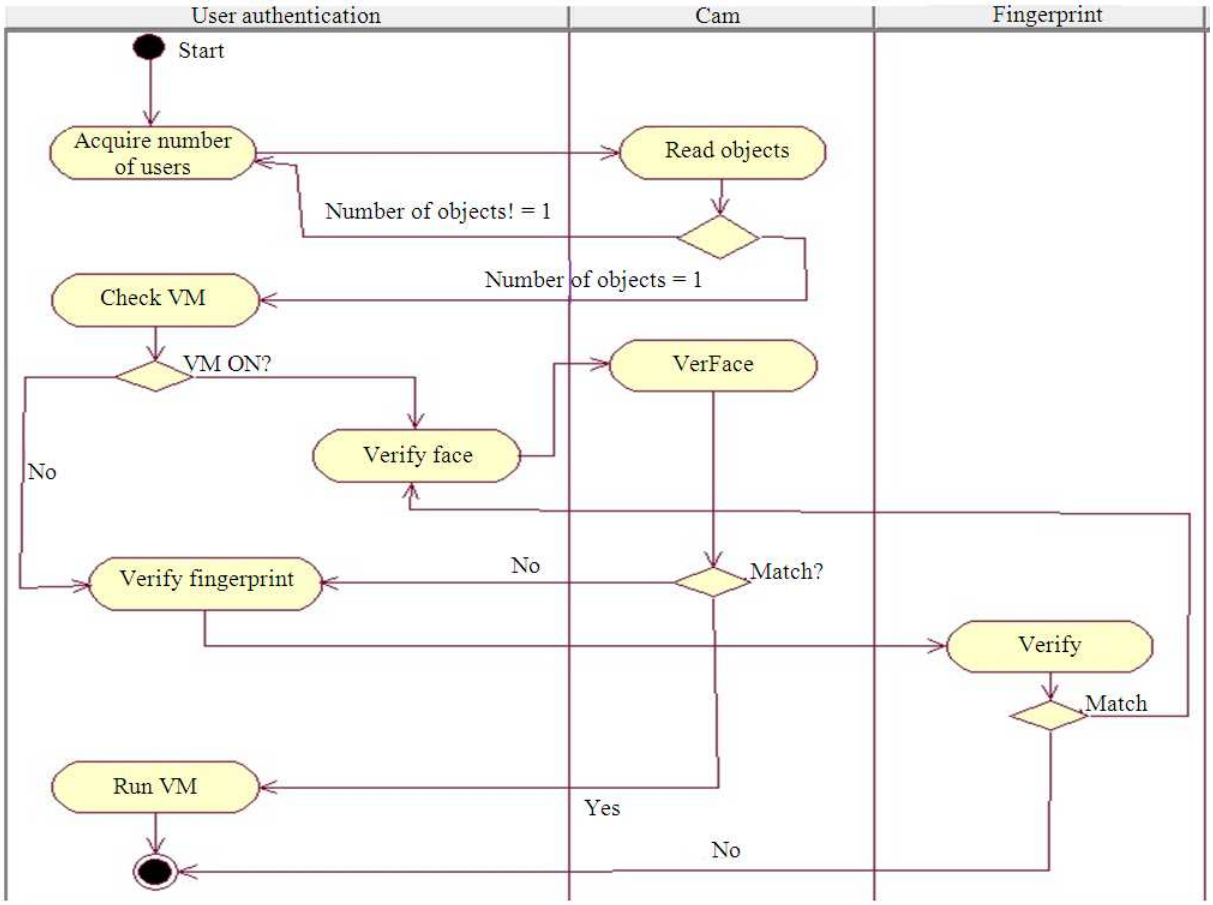

Fig. 4. System activity diagram for TPM-UAM

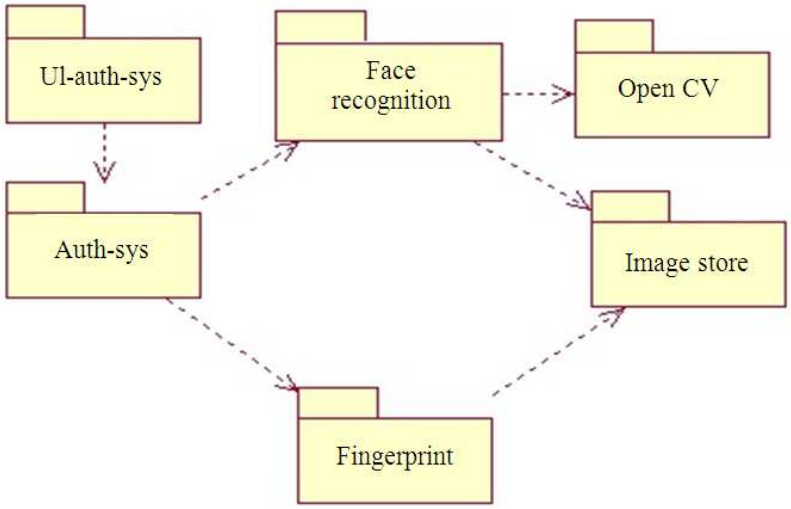

Fig. 5. Diagram of package system

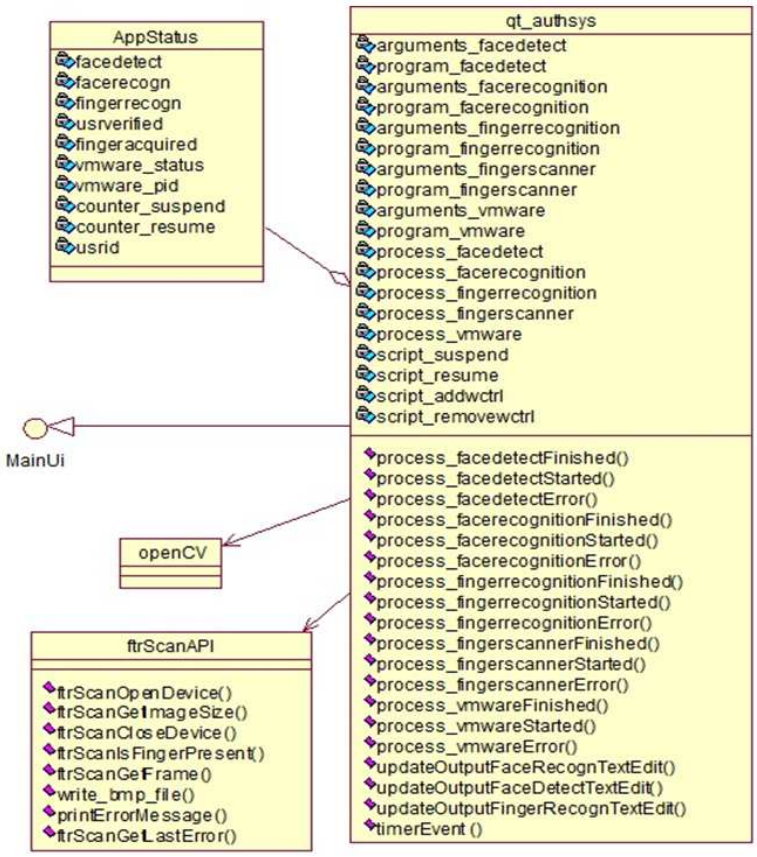

Fig. 6. System class diagram

The open CV package supports the Face recognition package, while the Fingerprint package contains the *ftrScanAPI* class and the Image Store package, which handles the processes of storing and restoring the images of the fingerprints and faces.

The system contains the following main classes, as shown in **Fig. 6**: Firstly, the *qt-authsys* is the main class in the system, where the processes for bringing the different parts of the system to work are included here, together with the main functions of bringing the face recognition to work and obtaining the results of the face recognition, which works in cooperation with the *open CV* package. This package contains the *haarclassifier* algorithm and other functions which are needed to make the face recognition process work and serve as required in the system. There are also some other processes here for communicating with the *ftrScanAPI* (the class responsible for handling the fingerprint scanner by the vendor of the scanner); Secondly, the *App Status* class works to confirm the status of the system and to store values on the status of the different components of the system to determine whether to keep the system running or to suspend the system for security reasons; Thirdly, the *ftrScanAPI* class controls the processes for handling the fingerprint scanner in such functions as opening the device, reading the fingerprint, checking the presence of the fingerprint, closing the device and writing the fingerprint image to an image file; Fourthly, the *Main IU* is the User Interface of the main system, where the inputs and outputs of the system will be running and where the different messages of the system will be displayed.

3.4. System Deployment

Based on the requirements of the system and the suggested design of the class and the package diagram, the deployment diagram represents the physical aspects of the TPM-UAM model. The deployment diagram, as shown in **Fig. 7**, shows the PC running with the Fedora 18 with Xen hypervisor. The TPM-UAM model will be

seated on top of this platform, while the camera and fingerprint scanner will be attached and connected to the system on the same platform. Also, as shown in **Fig. 7**, another platform with Windows 7, which will be referred to as a secure platform, will be installed later on the same machine to handle the TPM.

4. IMPLEMENTATION

4.1. System Setup

The Xen 4.3 hypervisor is used to create the required platforms. In this system, Xen has been installed on the top of the Linux system (Fedora 18). From the main menu of the boot loader, a choice has to be made to boot the Fedora 18 with the Xen hypervisor, so as to have the privilege of controlling the Linux kernel, as well as the ability to create another platform for use on the top of Xen. After installing Xen on the top of the Fedora, testing is performed by booting the system. Then, the Microsoft Windows 7 is installed as the second platform which will run the TPM on top of Xen.

4.2. Software Tools

To implement the desired system, the face recognition and detection techniques and the fingerprint detection and recognition have to be placed and implemented. To achieve that, the image processing library has to use the open CV (C++) as it has the benefits of:

- Face detection with a Cascade of Boosted Classifiers Based on Haar-like Features used from the open CV class (Cascade Classifier)
- Face recognition with Local Binary Patterns Histograms used from the open CV class (Face Recognizer)
- Fingerprint recognition with normalized correlation used from the open CV function (match Template)

The main system application is written in C++ using the Qt framework.

4.3. The User Interface

Once the system is booted and the user is facing the Fedora 18 OS, the user can choose to start the secure platform, if needed. Once the user clicks the link to start the secure platform, he will pull the trigger to launch the TPM-UAM system. The system starts to work immediately by counting the number of users in front of the PC (**Fig. 8a**). The system will test to ensure that there is only one user present, before moving on to the next step, which is to read a user's fingerprint to confirm the user's identity (**Fig. 8b**). Here the user has to scan his fingerprint at the fingerprint scanner device to allow the system to take a reading (image of the fingerprint). As the system obtains an image of the fingerprint, the fingerprint recognition process will determine if the database has an identical image of that fingerprint, which proves that the user is a registered user (**Figure 8c**) If the user is found, then the system will confirm that the user has been verified and shows the user's ID in the database (**Fig. 8d**).

Now, as the system recognizes the user's fingerprint, the system should find if the associated face with that fingerprint in the database matches the image taken for the face in front of the PC (**Fig. 8d**). When the system finds the match, the user gets verified and approved. Thus, the system will launch the second secure platform, which is Windows 7, that will launch the TPM.

4.4. Testing

A system test is designed to evaluate the operational efficiency and appropriateness of a system after it has been completely integrated and to verify that it meets the specified requirements. The requirements for a user authentication system are clearly identified through the TPM user authentication model, as shown in **Fig. 1** and the use case diagram in **Fig. 3**. Therefore, functional testing is conducted to confirm the functionality of the system and to prove that the system has met the desired requirements.

To confirm the functionality of the system, a functional testing was conducted by the researcher and five other users, who were asked to test the system based on given test case scenarios and then to write down the result of each test carried out by them. Each user was advised to test the system before and after enrolling in the system to confirm the behaviour of the system with the unauthorized users.

The functional testing for the specified test cases were as follows: Test 1 (Authorized user Start the Secure Platform), Test 2 (Unauthorized user Start the Secure Platform), Test 3 (Confirm the presence of single user), Test 4 (Detect changes in number of users), Test 5 (User identification), Test 6 (Detect User's Face.), Test 7 (Face Recognition), Test 8 (System's response to the absence of authorized user), Test 9 (System's response to the presence of another user), Test 10 (System's response to the return of authorized user) and Test 11 (System's response to the return of unauthorized user). The results of the tests are presented in **Table 3**.

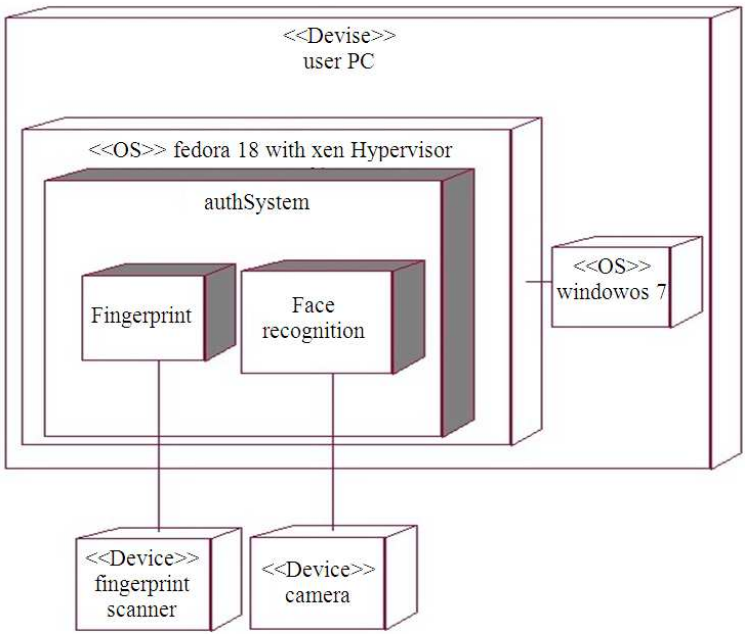

Fig. 7. System deployment diagram

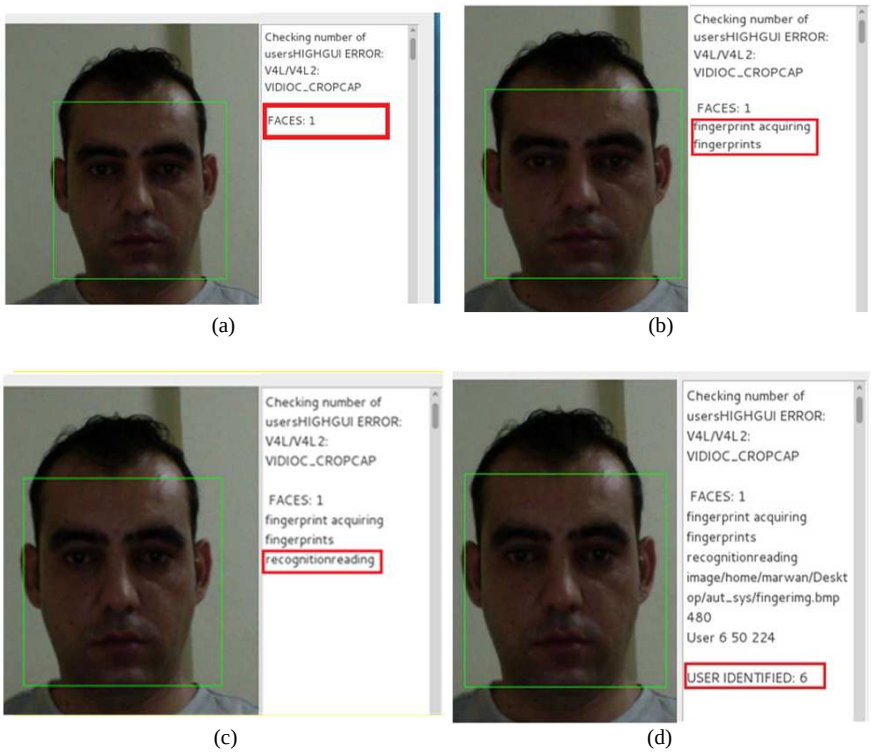

Fig. 8. (a). Verifying number of users, (b). Acquiring user fingerprint, (c). Fingerprint recognition process, (d). Fingerprint is verified and user ID found

Table 3. Results of functional testing

Users	Authorized user	Unauthorized user	Passing rate (%)
Test 1	√	-	100
Test 2	-	√	100
Test 3	√	-	100
Test 4	√	-	100
Test 5	√	-	100
Test 6	√	-	100
Test 7	√	-	100
Test 8	√	-	100
Test 9	√	√	100
Test 10	√	-	100
Test 11	-	√	100

All the test cases applied by the testers approved the functionality behind each function of the system and the system behaved as it was supposed to for all the tests.

5. RESULTS

The TPM-UAM model was presented and evaluated qualitatively using the focus group approach, where a group of experts evaluated the proposed model and confirmed the model efficiency (Alshar'e *et al*., 2014). In this paper we introduced the design and the Implementation of the TPM-UAM model. The TPM-UAM system was successfully developed and tested, where the design of the test cases focusses on the security aspects of the model as presented and evaluated by the focus group to prove the system's ability to protect the TPM. The system testing shows high resistance against physical attacks for the three aspects, which are, (1) to prevent direct interaction between unverified users with TPM where attacks such as in Evil Maid shall be eliminated, (2) to support the weak authentication of TPM and (3) to supervise the running TPM session and protect TPM in the run state.

6. DISCUSSION

This paper introduced the design and implementation of the TPM-UAM model, which proved to be a useful model in protecting the TPM. To implement the system, the open source image processing library from the open CV was used to ease the development of the Face recognition and Fingerprint process through the use of available and free libraries. The Xen hypervisor was installed on top of the Fedora 18 to support the virtualization and to create the desired multiple number of platforms.

A functional test was conducted to confirm the system's functionality and the results obtained from the test confirmed this as well as the proper response to the various situations described in the test cases.

The TPM-UAM model has been introduced as a safety guard to protect the TPM from physical attack. For the purpose of proving the efficiency of the TPM-UAM, a complete design and implementation was carried out to bring the model to life and to confirm the ability of the TPM-UAM to protect and secure the TPM and a platform with the TPM on it.

The work in this study is based on Xen virtualization, as the system has to be booted with a Xen hypervisor in order to be able to use the virtualization. It is highly recommended that the GRUB of the fedora system be edited such that it will be booted only by this option, "Fedora, with Xen hypervisor," so that the system will start directly with the Xen support. The benefit of this is that the system will not be booted if it is not passed by the TPM-UAM. Since the Xen modifies the kernel of the fedora, the TPM will never work only via the virtual platform that was built with the TPM-UAM and this makes the TPM more secure.

As this model focuses on the physical attacks of the TPM and to provide a solution to threats such as Evil Maid which compromise the TPM safety and security, the database of the biometric authentication is beyond the scope of this research paper, as the main concern is towards securing TPM and the biometric authentication is implemented to support the purpose of this research study at the user verification process session only of the model.

7. POSSIBLE THREATS AND CHALLENGES

As our model suggests to bring the use of TPM to virtual environment to prevent Evil Maid and related attacks, the model might be subjected to typical threats

toward virtual machines, such as security vulnerabilities in virtualization, where the most common and important threats were as follows (Reuben, 2007; Luo *et al*., 2011; Rehman *et al*., 2013).

7.1. Attack Between VMs or Between VMs and VMM

As the virtualization provide isolation which helps to protect processes and applications, isolation might be considered as a serious threat if it is not configured well or if the access control policy is not configured properly as well. This could result in an inter attack between virtual machines or Virtual Machines (VMS) and Virtual Machine Monitor (VMM). As a solution for this problem we recommend to limit the use of VMs to one VM only, which supposed to hold the TPM, also to make sure to configure the VM properly.

7.2. VM Escape

VM escape is a threat where the isolation between VM and the host is compromised. In case the VM escape happening, an application on the virtual machine could avoid the VMM and access the host machine which runs with root privilege, thus the application will run now with root privilege escaping from VM privilege.

This problem can be solved by properly configure host and guest interaction (Reuben, 2007). In our model VM escape is not supposed to threaten the VM, as we suggest the use of one VM and this VM will works only during the presence of the owner of the platform.

7.3. Virtual Machine Controlled by Host Machine

Host machine at the virtual environment considered as a control point where the host works to monitor a number of aspects related to virtual environment, such as start and shutdown and pause and restart of the VM; monitor and modify resources available for VM; monitor the applications running inside the VM if the rights is granted; and copy or view or even modify data stored at the virtual disks assigned for VM. Thus, the protection of the host is highly considered to keep VM secure. As a solution Reuben (2007) suggests proper isolation and configuration for the VM to avoid the host to act as a gate for the attacker.

7.4. Denial of Service

As the construction of VMs requires each VM to have specific amount of available resources, DOS attacks can use one of the VMs to consume all available resources. The consequences of this attack is that other VMs will not work and report denial of service due to the lack of the available resources to run the VM. Reuben (2007) suggests limiting the resources available for each VM to overcome this problem, where proper isolation with limited resources will prevent the occurrence of this problem.

Virtual Machine threats should critically considered and treated, as the presence of any VM related issues might compromise the security of the proposed model and in consequently affect the security of TPM. Therefore, a decent knowledge regarding threats against VM and proper configuration for the VM must be present and implemented to avoid the VM threats and then preserve the security of the model. The listed above threats can be easily overcome with the proper guiding and configuration of the VM.

Implementing the TPM-UAM model shall force high security level within the machine, which holds the TPM and runs with what the model suggests and requires. The main challenge which needs to be addressed is the system acceptance. While developing the model two main considerations needed to be carefully thought of, which are, security and user acceptance. The model focusses mainly on the security perspective thus, user acceptance for the model might be considered as the main challenge, as the model requires the presence of the user for the VM which holds the TPM all the time to keep it working. If the authorised user is absence and system cannot detect him or her, the VM platform will enter in a pause state and will return to work again only when the authorised user return and get verified successfully, as we suggests earlier. Monitoring running session of TPM is as important as confirming the identity of the authorised personnel at the logging time to TPM, thus it is important to protect TPM against attacks at this critical time. Therefore, holding the user for some time until certain work is done is important to be accepted and tolerated by the user as it will result in preventing the TPM being attacked and as the platform will be paused, still the user can leave and return later and the system will proceed the work from last paused point regularly.

8. CONCLUSION

The implementation of the TPM-UAM model has been presented in this paper. The implementation of this model has shown the use of the virtualization concept for the creation of multiple platforms on the same machine. The first platform is used to authorize users and the second one is to run the TPM. User privacy and confidentiality are protected by monitoring the presence

of the authorized user all the time. Biometric authentication techniques are used to authenticate users in terms of identity and authority to use the TPM.

The implementation of the TPM-UAM model shows the ability to secure the platform containing the TPM by forcing users to pass the authentication process at the authentication platform. Only when the user has passed the authentication can the secure platform, which contains the TPM, be run. Two main facts have been used to assess whether the model works perfectly. Firstly, the nature of the TPM which supports only a single owner, which in this case is brought to the virtual platform; and secondly, the Xen modifies the kernel of the OS, which will be detected by the TPM. Since the TPM will not work on a modified kernel, thus it can be brought to work only on the virtual platform.

The functional testing of the model has approved the system's functionality, where the prototype could respond to various situations and tasks as planned. For future work, an evaluation process for the model is being developed by performing an expert review of the system.

For future work, we could consider securing the biometric database to enhance the overall security of the model.

9. ACKNOWLEDGMENT

The researchers would like to thank the Ministry of Higher Education of Malaysia for financial support of this project under the FRGS grant, FRGS/1/2011/SG/UKM/03/4.

9.1. Author's Contributions

Marwan Ibrahim Alshar'e: Propose the main model, designing experiments, contribute to the writing of the manuscript.

Rossilawati Sulaiman: Brainstorming the model and experiments, contribute to the writing of the manuscript.

Mohd Rosmadi Mokhtar: Brainstorming the model and experiments, contribute to the writing of the manuscript.

Abdullah MohdZin: Organize the study, brainstorming the model and experiments, contribute to the writing of the manuscript.

9.2. Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

10. REFERENCES

Abels, T., P. Dhawan and B. Chadrasekaran, 2005. An overview of xen virtualization. Dell Power Solutions, 8.

Alshar'e, M.I., R. Sulaiman, M.R. Mukhtar and A.M. Zin, 2014. A user protection model for the trusted computing environment. J. Comput. Sci., 10: 1692-1702. DOI: 10.3844/jcssp.2014.1692.1702

Bower, T., 2010. Experiences with virtualization technology in education. J. Comput. Sci. Colleges, 25: 311-318.

Chaganti, P., 2007. Xen Virtualization: A Fast and Practical Guide to Supporting Multiple Operating Systems with the Xen Hypervisor. 1st Edn., Packt Publishing, Birmingham, ISBN-10: 1847192483, pp: 148.

Dalton, C.I., D. Plaquin, W. Weidner, D. Kuhlmann and R. Brown *et al*., 2009. Trusted virtual platforms: A key enabler for converged client devices. ACM SIGOPS Operat. Syst. Rev., 43: 36-43. DOI: 10.1145/1496909.1496918

Gareau, J., 1998. Advanced embedded X86 programming: Protection and segmentation. Embedded System Programming.

Götzfried, J. and T. Müller, 2014. Analysing android's full disk encryption feature. J. Wireless Mob. Netw. Ubiquitous Comput. Dependable Applic., 5: 84-100.

Hagen, W.V., 2008. Professional Xen Virtualization. 1st Edn., Wiley Publishing, Hoboken, ISBN-10: 047028918X, pp: 500.

Jain, A., L. Hong and S. Pankanti, 2000. Biometrics: Promising frontiers for emerging identification market. Commun. ACM, 43: 91-98.

Jain, A.K., A. Ross and S. Prabhakar, 2004. An introduction to biometric recognition. IEEE Trans. Circuits Syst. Video Technol., 14: 4-20. DOI: 10.1109/TCSVT.2003.818349

Khairwa, A., K. Abhishek, S. Prakash and T. Pratap, 2012. A comprehensive study of various biometric identification techniques. Proceedings of the 3rd International Conference on Computing Communication and Networking Technologies, Jul. 26-28, IEEE Xplore Press, Coimbatore, pp: 1-6. DOI: 10.1109/ICCCNT.2012.6396051

Khushk, K.P. and A.A. Iqbal, 2005. An overview of leading biometrics technologies used for human identity. Proceedings of the Student Conference on Engineering Sciences and Technology, Aug. 27-27, IEEE Xplore Press, Karachi, Pakistan, pp: 1-4. DOI: 10.1109/SCONEST.2005.4382869

Klenk, A., H. Kinkelin, C. Eunicke and G. Carle, 2009. Preventing identity theft with electronic identity cards and the trusted platform module. Proceedings of the 2nd European Workshop on System Security, Apr. 01-03, Nuremberg, Germany, ACM New York, pp; 44-51. DOI: 10.1145/1519144.1519151

Liu, S. and M. Silverman, 2001. A practical guide to biometric security technology. IT Professional, 3: 27-32. DOI: 10.1109/6294.899930

Luo, S., Z. Lin, X. Chen, Z. Yang and J. Chen, 2011. Virtualization security for cloud computing service. Proceedings of the International Conference on Cloud and Service Computing, Dec. 12-14, IEEE Xplore Press, Hong Kong, pp: 174-179. DOI: 10.1109/CSC.2011.6138516

Müller, T., H. Spath, R. Mäckl and F.C. Freiling, 2013. Stark. In: Financial Cryptography and Data Security, Sadeghi, A.R. (Ed.)., Springer Berlin Heidelberg, ISBN-10: 978-3-642-39883-4, pp: 295-312.

Prabhakar, S., S. Pankanti and A.K. Jain, 2003. Biometric recognition: Security and privacy concerns. IEEE Security Privacy, 1: 33-42. DOI: 10.1109/MSECP.2003.1193209

Ray, E. and E. Schultz, 2009. Virtualization security. Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies, Apr. 13-15, ACM, USA, pp: 42. DOI: 10.1145/1519144.1519151

Rehman, A., S. Alqahtani, A. Altameem and T. Saba, 2013. Virtual machine security challenges: case studies. Int. J. Machine Learn. Cybernetics, 5: 729-742. DOI: 10.1007/s13042-013-0166-4

Reuben, J.S., 2007. A survey on virtual machine security. Security of the End Hosts on the Internet, Seminar on Network Security Helsinki University of Technology.

Rutkowska, J. and A. Tereshkin, 2009. Evil Maid goes after True Crypt.

Spector, S. and Xen, 2011. Why Xen.

TCG, 2010. Black hat conference report about TPMs: TCG in Action.

Zajkowska, J., M. Kondrusik, S. Grygorczuk, B. Skotarczak and B. Wodecka *et al*., 2007. Molecular and serological diagnosis of borrelia burgdorferi infection among patients with diagnosed *erythema migrans*. Ann. Agric. Environ. Med., 14: 209-213.