Paper

Drive Encryption and Secure Login to a Secure Workstation for Special Applications

Marek Małowidzki, Tomasz Dalecki, and Michał Mazur

Military Communication Institute, Zegrze, Poland

Abstract—We discuss the problem of a secure login to a virtualized workstation. For increased security, the workstation's hard drive is encrypted. During the startup, a decryption password to the drive must be entered by a user. We propose a solution that involves mutual authentication between the workstation and the user and ensures the password may be entered securely.

Keywords—drive encryption, evil maid, Linux, secure login, TPM, virtualization, Xen.

1. Introduction

Our work is a part of a project aiming at a development of a virtualization-based secure workstation, designed for running classified software or processing sensitive data. An important component of the overall security is hard drive encryption (portable media encryption is considered as well). Also, a workstation may be regarded as secure only if it runs original, unmodified software. Assuring this is easier in a physically protected environment; in case of portable computers, however, it poses a challenging problem. In the paper, we propose a solution that allows to combine hard drive encryption with a trustful boot process, preventing risk of software tampering.

The paper is organized as follows: First, we briefly describe the project. Then, we overview drive encryption software. The subsequent part of the paper defines the problem of secure logon, and discusses the boot process in details. We assess the security level offered by our solution, then propose further work and end the paper with conclusions.

2. Secure Station for Special Applications

The work described in the paper has been performed as a part of Secure Workstation for Special Applications[1] project, implemented by a consortium of four companies (the Military University of Technology, the Research and Academic Computer Network (NASK), Filbico, and the Military Communication Institute). The project's goal is to develop a virtualization-based workstation, designed for special applications. It is assumed that each virtual machine (except for the "control" domain 0) processes data from a different domain – by "data domain" we understand either a classification level (unclassified, restricted, etc.) or a purpose (e.g., a financial system, a human resources system, an IT software project), and because of that the data and related processing activities should be separated. This separation, together with increased security, provided by a hypervisor, is the most important reason for the selection of a virtualized environment.

By "special applications" of a workstation we mean its typical use in a classified system, e.g., in a chancellery for storing sensitive documents or as a machine for running classified software. While the focus of the project is put on desktop workstations, we also consider mobile computers (laptops). At this stage of the project, we do not consider networking (and related security risks). Generally, we silently assume the workstation is a standalone system.

After a review of open-source hypervisors that could be employed by the project, the consortium members selected Xen [1] (the version is 4.1). KVM [2] was another strong candidate. The final decision was made mainly on the basis of market presence and the significance of research projects related to each of the two hypervisors, and we believe in these two areas Xen evidently prevails.

3. Drive Encryption for Linux

The Linux distribution used in project is RedHat Enterprise 6.1. The default encryption system for this distribution is dm-crypt [3]. Our evaluation of dm-crypt confirmed that it fulfills all of our requirements, listed below:

- widely used and well-documented solution,
- open-source (or, with a friendly license),
- capable of full system encryption,
- easy to use in scripts,
- using standard format for encrypted volume (LUKS [4]),
- using a state-of-the-art encryption algorithm (AES).

[1]The project "Secure Workstation for Special Applications" has been founded by the Polish Ministry of Science and Higher Education under grant no. 0140/R/T00/2010/11.

Other solutions, namely, TrueCrypt [5] or eCryptfs [6], were also an option, although we rejected them as an unnecessary complication.
The dm-crypt (device mapper crypto target) package provides transparent encryption and decryption of block devices. It creates a logical device (e.g., `/dev/mapper/sda2crypt`) for each file or partition (e.g., `/dev/sda2`) we would like to encrypt. Reading from such a logical device involves a decryption operation performed in a background, while writing data through the device results in data encryption.

4. The Actual Problem: Secure Logon

In order to decrypt the drive, we need to pass a password (or, a key) to drive encryption software. Of course, as the password cannot be stored within the workstation (at least, in a plain form; but if it is encrypted, then we need a password-to-password), it must be entered by a user during workstation startup. This leads to an important question whether the workstation's environment can be trusted – that no malware is waiting to intercept the password (and store it somewhere for further retrieval, as in the evil maid case [7], [8]).
As one can see, for our secure workstation, we only selected an existing encryption package and had no plans to develop dedicated software, nor modify the selected one. However, the actual problem we face here is how a user could securely provide the password, or, how we could assure that the workstation has not been modified in any way. That is to say, we need to assure a secure logon. Or, we require the workstation to authenticate itself to a user before a user authenticates in the workstation [8].
The above mentioned evil maid attack is less likely in a secure (physically protected) external environment, where a workstation is intended to be installed and used. However, as we have mentioned previously, our project also considers a "mobile workstation" in the form of a laptop, and is trying to deliver a flexible solution for both cases.

5. Requirements

Drive encryption, correlated with secure logon, should fulfill a number of requirements. Some of them are more obvious, and some sound less apparent. The paramount requirements are as follows:

- efficient and effective data encryption, resistant to crypto attacks;
- workstation configuration (virtual machines and users) protected from disclosure;
- workstation environment (both software and data) shielded from tampering.

Additional, less apparent requirements are listed below:

- authentication of the workstation to a user: the workstation should convince the user that its environment is original and untampered;
- authentication of a user in the workstation, preferably, involving additional hardware and/or biometric elements;
- support for multi-access: we assume a number of user accounts, even in the case of a laptop;
- effective means to limit access to virtual machines for users that have no accounts there;
- easy and effective means to withdraw a user's access to the workstation.

The scope of encryption includes the whole hard drive without the content of `/boot` directory of the host system (domain 0). This directory contains the boot image, the initial file system (`initramfs` [9]) and user account data, packed and encrypted as a single *blob* file (refer to the next Section for details).

6. The Boot Process Step-By-Step

During the boot process, a number of software pieces are executed in a sequence (or, a *chain*). The boot may be considered secure if all the pieces are original (not modified by an attacker). To ensure this, every executing element measures its successor in the chain before passing the control to it, and writes the measurement to a Trusted Platform Module (TPM [10]) register. The first element in the chain, the Core Root of Trust for Measurement (CRTM) is a hardware-protected boot block code (e.g., the BIOS boot block code). The CRTM is considered trustworthy. The last element in our case is `initramfs`, measured by a bootloader. If all the measured values are correct and match the values of the data the TPM has *sealed* – i.e., when workstation boot software has been prepared and configured [11], the TPM enters the desired state and is able to perform cryptographic operations.
Before we proceed, we describe the content of a USB stick, which acts as a hardware key, and is prepared for each user during account creation. A BESTSTICK (BEST is an acronym from the Polish title of the project) contains four files, residing in an archive encrypted by the TPM:

- `login`: the file contains the user name;
- `password`: this is the random part of a password (to password, see below), generated using the `/dev/random` device; note that `/dev/random`, unlike the `/dev/urandom` pseudo-random device, may be considered as a true random generator, as it uses environmental "noise" (e.g., mouse movements or keyboard strokes);
- `phrase`: this is the authentication phrase;

– `image`: this file contains the authentication image, also selected by a user during the account creation. In practice, the image should be copied from another portable storage medium during the stick preparation.

The boot process, after `initramfs` has been loaded and the control passed to its main executable – `/init`, proceeds in the following steps:

1. A RAM drive is created and mounted. All further file operations are performed using the RAM drive.
2. A user's BESTSTICK is mounted and the archive it contains is copied to the RAM drive.
3. The archive is decrypted using the TPM, and the individual files are extracted. The stick is no more needed, so it is unmounted.
   Note that the TPM will switch to the desired state only if the workstation boot software chain (from CRTM to `initramfs`) has not been modified (also assuming that the software is bug-free). Otherwise, the attempt to unseal (decrypt) the secret will fail and the subsequent steps will not execute.
4. Now, the splash screen (plymouth [12]) is configured to display the phrase and picture, and the user is asked for his part of drive encryption password (`userpass`) – see Fig. 1.
5. The file containing encrypted password to the drive, `username.dom0.aes`, is extracted from the encrypted archive `/boot/best/users`; again, the TPM chip performs the decryption.
6. The file *`username`*`.dom0.aes` is decrypted using the AES algorithm, with a password composed of two parts – the random part and the user-provided part: (`userpass || password`).
7. The password to the hard drive is passed to `cryptsetup`; the RAM drive's content is overwritten with zero bytes, and then gets unmounted.
8. The user can log in to the workstation.

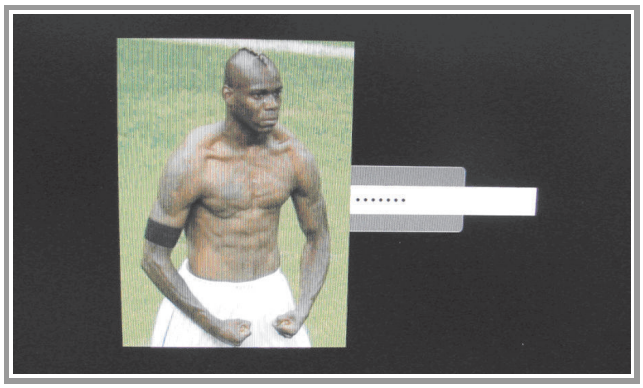

Fig. 1. The login screen (the phrase is visible after switching to console view).

In case of a "virgin" run (with no configured users), the above procedure is skipped and a special script is executed that allows to create the first (administrator) account.
The main components of the process are summarized in Fig. 2.

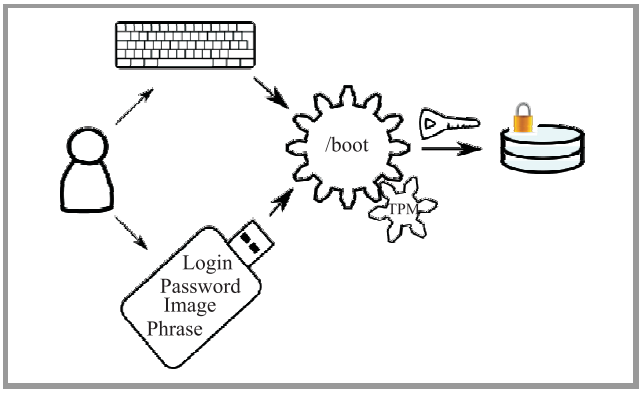

Fig. 2. Key components of the login process.

The process is executed by quite a large number of shell scripts. Most of them are embedded into `initramfs` (we use `dracut` [13] to automate `initramfs` generation). Some additional scripts perform administrative tasks (e.g., create a new user account and configure a USB stick, etc.), some are used for testing. Additional software installed on the workstation included TrustedGRUB [14] (the GRUB bootloader with TPM support), and two packages providing access to TPM: TrouSerS [15] and tpm-tools.
Note that the final effect we have reached is not completely satisfying. It is quite difficult to manage the display of the picture (a large picture happens to partially cover the password area, which looks bad). Also, we would prefer the phrase to be shown directly on the login screen. Unfortunately, while it was possible to modify and recompile `plymouth` code, we failed, for an unknown reason, to make it work after re-installation. Perhaps it requires some specific approach at some step.

7. Drive Encryption Revisited

We performed a broad set of tests to assess the efficiency of drive encryption of our Seagate Constellation 1000 GB HDD drive. A few jumbo files and a lot of small files were copied between encrypted and unencrypted partitions. The tests required some care as the system "cheats" by caching data read from the drive (for example, we rebooted the system often enough to prevent this behavior). Also, during the experiments we observed that the physical location on the drive where data are read from (or, written to) significantly influences performance, probably due to differences in drive access times. Moreover, these times often dominate the encryption. As a result, we can issue a conclusion that, while the overhead of encryption is difficult to measure precisely, it is quite small and does not lead to a negative user experience. We also performed tests for

a double encryption case, assuming that virtual domains may use additional encryption – see Section 8. This time, the performance degradation was observable but the overall system performance was still regarded as acceptable.

8. Security Assessment

In this Section, we assess the security of our architecture – the strong and weaker points and possible improvements. Note that we generally assume the worse usage scenario of a laptop stored and used in a (potentially) insecure environment.

CRTM and TPM. First, we rely on the Core Root of Trust Management and the Trusted Platform Module – that the whole boot process could be trusted. While some first successful attacks on TPM have been reported [16], they are enormously time- and resource-consuming (one could argue that with sufficient time and unlimited technical means every security mechanism could be compromised), and it seems we can assume that boot elements are secure enough.
At the same time, we assume the whole **boot software chain** is correct and immune to attacks (such as buffer overflows). Otherwise, an attacker could modify its behavior to prepare subsequent hostile actions while supplying original hash values to the TPM.

Phrase. This is an additional and quite a weak mechanism as the phrase could be intercepted relatively easily (e.g., by a hidden camera in a hotel room). Having said that, we can add that the mechanism could be augmented using a set of phrases, with a user selecting a phrase to display using some identifier (a phrase number or a few first letters). Assuming the user is careful enough and changes phrases during subsequent logins, the mechanism would be greatly strengthened.

Picture. At present, we assume it is much more difficult to "steal" the picture (e.g., with a precise photo). Unfortunately, even an imperfectly copied picture could suffice if the user is not careful enough (he might not notice some subtle differences at the pixel level). As above, a set of pictures could be employed to make an attack harder to prepare.

Password typing. A simple attack could be performed using a hidden camera in order to observe the password during typing. For increased security, if a user is able to type without looking at the keyboard, some form of a keyboard cover would prevent the snooping.

Hardware key (USB stick). The key must not be kept together with the workstation. The secrets (phrases and pictures) used for the workstation to authenticate itself to a user are decrypted automatically at startup, so they are revealed to any user, legitimate or not, who is powering the workstation with the hardware key present. There could be an additional (preliminary) password applied to decrypt the stick's content (together with TPM, of course – that is, both the preliminary password and TPM would be necessary to decrypt the stick). This would yield increased security at the cost of increased complexity of the login process.

Also, a plain USB stick may easily be copied (without leaving any trace), which also constitutes a security hole. Thus, we consider a smart card as a replacement for the stick.

Biometrics. Most modern laptops are already equipped with some sort of biometry device (e.g., fingerprint scanner), which would help in user identification.

Access withdrawal. This is relatively easy, as it is sufficient to remove a user's account by deleting the `username.dom0.aes` file (containing the encrypted password to domain 0). Even if a user refuses to return the USB stick (or, makes a copy of it), there is no way the user can log in to the workstation, and the stick becomes unusable.

Workstation configuration: users. User accounts (their presence and number) are protected; their data are encrypted.

Workstation configuration: virtual domains. Domains could be stored either as files within the encrypted file system, or using separate (encrypted) partitions (see the next paragraph). In the latter case, it seems that the number of domains could be observed but, having a sufficiently capacious hard drive, one could configure a large (sufficient in all scenarios plus some spare ones) number of partitions; some of them would be used by actual virtual domains, while other would contain random data. In this way, the presence and the number of domains would be concealed. However, this is not the full story yet: An attacker could observe changes in raw data and infer which partitions are used and which lie dormant, so some additional effort (altering the bytes from time to time) would be necessary.

Virtual domains encryption. Although our proposal concerns logging to domain 0, we assume that user domains

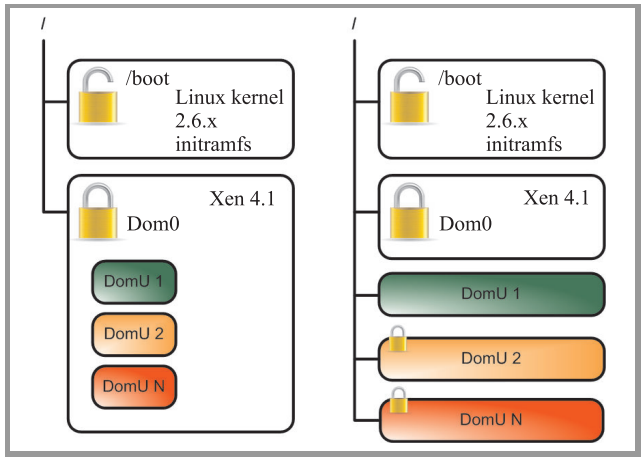

Fig. 3. Virtual domains as files within domain 0 (left) or separate drive partitions (right).

will be additionally encrypted, each using a separate password. The reason is that in case a malicious user (or, a malware that infected one of his virtual machines) is able to successfully break into domain 0, he will not be able to immediately attack inactive domains (lying dormant on the drive), as he does not know the decryption password. This is not a perfect protection mechanism (the attacker could install password interception malware), but it makes the attack on the other domains harder and more time-consuming.
We consider two possible approaches. The virtual domains may reside in files within the domain 0, or be stored in separate disk partitions (see Fig. 3).

Summary. As we have shown above, the logon process offers a reasonable level of security. This level could be increased by some additional mechanisms (multiple phrases, additional password, etc.). However, this also results in increased complexity. As [17] demonstrates, a user (human being) is the weak point; if a user finds the login process as unacceptably complex, he will also discover a way to simplify it in a way that definitely will also relax the security. Thus a sensible compromise must be found.
For an in-depth discussion of security of a system with encrypted drive in the context of the evil maid attack, possible prevention methods and ways to bypass them, refer to [7], [8], [18].

9. Related Work

The work [8], [19] concerns the same problem. We learned a lot from that work (especially the discussion of possible attacks), although technically our solution is different and offers some additional features required in our project – with multi-access (support for multiple user accounts) being probably the most important issue. In our case, the stick is not bootable; it serves for authentication purposes only. Other differences between [8], [19] and our solution are listed in Table 1. Note that by showing the comparison, we are not going to argue that our work is better in some way. We believe that it should be treated as a sort of continuation, and the table is presented for a reader's convenience.

Table 1
Comparison of features between [8] and our work

	Anti-evil maid	Our solution
Location of system authentication data	stick	stick
System authentication data	phrase	phrase and picture
Ability to use SRK password	yes	no
Part of the disk password on the stick	no	yes
Part of the password stored locally	no	yes
Location of the boot files (kernel, initramfs, ...)	stick	hard drive

10. Future Work

The main issues we would like to take into account in future involve the application of smart cards (as a replacement for the current USB sticks) and experiments with on-board biometric devices (in laptops). Also, we would like to make yet another attempt to modify `plymouth` and make it display both phrase and picture in a neat way.

11. Conclusions

In the paper, we have presented our proposal to implement a secure login to a Linux-based workstation with an encrypted hard drive. We believe it offers a reasonably high level of security; this level may additionally be increased at the cost of making things more complex (and, probably, less tolerated by a user). As a final remark we would like to note that our work remains valid also for a non-virtualized Linux system.

Acknowledgments

The authors would like to thank Joanna Rutkowska for a review and many insightful comments, which helped to improve the quality of the paper.

References

[1] "Xen" [Online]. Available: http://xen.org/

[2] "Kernel Based Virtual Machine" [Online]. Available: http://www.linux-kvm.org/page/Main_Page

[3] "Dm-crypt" [Online]. Available: http://www.saout.de/misc/dm-crypt/

[4] "LUKS + dm-crypt" [Online]. Available: http://code.google.com/p/cryptsetup/

[5] "TrueCrypt" [Online]. Available: http://www.truecrypt.org/

[6] "eCryptfs" [Online]. Available: http://launchpad.net/ecryptfs

[7] "The Invisible Things Lab's blog: Joanna Rutkowska: Evil Maid goes after TrueCrypt!" [Online]. Available: http://theinvisiblethings.blogspot.com/2009/10/evil-maid-goes-after-truecrypt.html

[8] "The Invisible Things Lab's blog: Joanna Rutkowska on Anti-Evil Maid" [Online]. Available: http://theinvisiblethings.blogspot.com/2011/09/anti-evil-maid.html

[9] "Initramfs" [Online]. Available: http://en.gentoo-wiki.com/wiki/Initramfs

[10] "Trusted Computing Group, TPM Main Specifications" [Online]. Available: http://www.trustedcomputinggroup.org/resources/tpm_main_specification

[11] "Trusted Computing: TCG proposals" [Online]. Available: http://www.cs.bham.ac.uk/~mdr/teaching/modules/security/lectures/TrustedComputingTCG.html

[12] "Plymouth" [Online]. Available: http://www.freedesktop.org/wiki/Software/Plymouth

[13] "Dracut" [Online]. Available: https://dracut.wiki.kernel.org/index.php/Main_Page

[14] "TrustedGRUB – How does it work?" [Online]. Available: http://projects.sirrix.com/trac/trustedgrub/wiki/HowDoesItWork

[15] "TrouSerS: The open-source TCG Software Stack" [Online]. Available: http://trousers.sourceforge.net/

[16] "The H Security – Hacker extracts crypto key from TPM chip", 10 Feb. 2010 [Online]. Available: http://www.h-online.com/security/news/item/Hacker-extracts-crypto-key-from-TPM-chip-927077.html

[17] K. D. Mitnick, W. L. Simon, *The art of Deception: Controlling the Human Element of Security*. Wiley, 2002.

[18] "Bruce Schneier's blog: Schneier on Security: Evil Maid Attacks on Encrypted Hard Drives" [Online]. Available: http://www.schneier.com/blog/archives/2009/10/evil_maid_attac.html

[19] "Joanna Rutkowska: Anti-Evil Maid installation script" [Online]. Available: http://git.qubes-os.org/?p=joanna/antievilmaid.git;a=blob;f=antievilmaid_install

Marek Małowidzki received his M.Sc.Eng. from Warsaw University of Technology. At present, he works at the Military Communication Institute where he carries out research on IT systems integration and security, and finalizes his Ph.D. on modeling traffic in TCP/IP networks. His main professional interests include distributed systems and technologies, programming languages and technologies, and TCP/IP networks.
E-mail: m.malowidzki@wil.waw.pl
Military Communication Institute
Warszawska st 22A
05-130 Zegrze Płd., Poland

Tomasz Dalecki received his M.Sc.Eng. from Warsaw University of Technology. He has been working at the Military Communication Institute since 2003. His research interests cover management systems design and implementation and security in IP-based systems.
E-mail: t.dalecki@wil.waw.pl
Military Communication Institute
Warszawska st 22A
05-130 Zegrze Płd., Poland

Michał Mazur received his M.Sc.Eng. from Military University of Technology, Warsaw. At present, he works at the Military Communication Institute where he is a programmer. His main professional interests include .NET and Java programming platforms as well as others programming languages and technologies.
E-mail: m.mazur@wil.waw.pl
Military Communication Institute
Warszawska st 22A
05-130 Zegrze Płd., Poland