

STRONGER AUTHENTICATION IN E-COMMERCE: HOW TO PROTECT EVEN NAÏVE USER AGAINST PHISHING, PHARMING, AND MITM ATTACKS

C. Latze and U. Ultes-Nitsche
University of Fribourg

ABSTRACT

Phishing, pharming and MITM attacks, i.e. the theft of user credentials, are a major threat to e-commerce applications. As soon as the attacker manages to talk a user into revealing his/her credentials needed to access an e-commerce application (e.g. user name, password, transaction number (TAN) in case of e-banking applications), the user's account is open to any kind of (financial) transaction by the attacker. In this paper, we propose using the trusted platform module (TPM) — a piece of hardware which will be built into all computers shipped in the near future — for ensuring both an e-commerce application's integrity and binding user authentication to user credentials and the usage of specific hardware during the authentication process. By doing so, strong authentication is achieved (something one knows is combined with something one possesses physically), which renders phishing attacks unsuccessful as the phisher will not be in possession of the required hardware and therefore getting user credentials will not open the e-commerce account for exploitation.

KEY WORDS

e-commerce, TPM, phishing, pharming, MITM

1. Introduction

Phishing, pharming and MITM attacks, i.e. the theft of user credentials, are a major threat to e-commerce applications [3]. As soon as the attacker (phisher) manages to talk a user into revealing his/her credentials needed to access an e-commerce application, the user's account is open to any kind of (financial) transaction by the attacker. Clearly, the problem is that user credentials usually are intangible, i.e. something the user knows (e.g. user name, password, transaction number (TAN) in case of e-banking applications). If the attacker possesses these intangible credentials, the e-commerce system will not be able to distinguish attacker and authorized user. There are several solutions, which are technically mature to avoid such problems, but almost all of them might be easily circumvented accidentally by unexperienced users as shown in Section 3. Therefore, the aim of this paper is to provide a solution, which reduces the risk of accidentally caused security problems.

In this paper, we propose using the trusted platform module (TPM) — a piece of hardware which will be built into all computers shipped in the near future — for ensuring both an e-commerce application's integrity and binding user authentication to user credentials and the usage of specific hardware. As the TPM will be included in the computers, it is a more comfortable device than smart cards, which require additional hardware to be connected. The TPM can seal information and process data using sealed information without making the sealed information accessible. Therefore a hierarchy of certifications and integrity tests is possible, from the BIOS (basic input/output system) during system start up to application integrity tests during application invocation. This first observation enables one to use integrity tested application for e-commerce. Secondly, strong authentication can be achieved using TPM-sealed private keys whose public counterpart is registered with the merchant: something one knows will be combined with something one possesses physically (the information is physically bound to the TPM), which renders the above mentioned attacks unsuccessful as the attacker will not be in possession of the required piece of hardware. As one does not always use one's machine at home, it is also possible to register portable devices such as a cell phone with the e-commerce provider, and the cell phone could be integrated in a strong authentication process.

Section 2 gives an overview over the three different types of attacks, we mention in this paper, followed by a Section over related work. Section 4 then gives a brief introduction to the TPM. Then, in Section 5, ensuring application integrity using the TPM is discussed. Section 6 finally presents TPM-supported strong authentication. The paper ends with some concluding remarks.

2. Phishing, Pharming and MITM attacks

Phishing attacks describe attacks, where an attacker tries to steal the credentials of an user using social engineering methods. This might be for instance an e-mail, where the user is asked to confirm her credentials on a special website, where the link can be found in the e-mail. This website is usually managed by the attacker, who may easily log the credentials.

Pharming is some sort of extension to classical phishing. It includes the manipulation of DNS servers or the user's host file to redirect the webbrowser to a malicious side. Again, the attacker may simply log the user's credentials given to this side.

Man-in-the-Middle (MITM) attacks are more sophisticated. The attacker is located between user and the legitimate server and snoops the traffic. On unsecured connections, such an attack is very easy, since neither the user nor the server will recognize the attacker. On a secure connection, the attacker has to establish a secure connection between herself and the user and herself and the server. An experienced user will recognize immediately, that she is connected to the wrong server, but a naïve user will not recognize any differences. The attacker can then store the user's credentials on her own machine and send them additionally to the correct server. That way, all the transactions, the user wants to do on the real server, will also be done, which avoids that the user becomes suspicious.

3. Related Work

Usually, the Secure Sockets Layer protocol (SSL) is used to authenticate a server. Using SSL, the web browser displays a certificate, which states that the server, the user has connected to is really the server, it pretends to be. But for naïve users, just displaying certificates and warnings in case of invalid certificates is not enough. They will simply accept everything, which leads to easily deployable Man-in-the-Middle (MITM) attacks. SSL provides optional mutual authentication, where the server has to verify the client additionally. The first problem with mutual authentication is that most of the naïve users are not able to retrieve their own certificates. Additionally, as long as the server does not have a list of clients, which are allowed to connect, it cannot decide whether the client is really the client, it pretends to be or whether it is an attacker, who phished the credentials from a legitimate client.

To help even unexperienced users, several solutions have emerged. They range from bigger warning messages [6, 2, 5] to better passwords [11], over SSL extensions [9] to the usage of trusted devices [4, 10]. But none of these solutions provides as much security as we provide.

In [6], the authors propose to extend the web browser with a so called "trusted credentials area". Such an area is a fixed part of the browser window, which displays authenticated credentials. The authors assume, that the problem of SSL is the textual representation of the server certificates, which makes it so complicated for naïve users. Representing certificates with easy to recognize logos should solve the problem according to [6]. In our opinion, this does not solve the problem, since it is still possible that users ignore the certificates. Furthermore, this approach is highly vulnerable to a proxy MITM. An attacker may simple use a valid certificate, which will result in a valid logo and phish the user's credentials.

Another approach in the class of "bigger warning messages" is [2], where the authors deployed a browser plugin called *Spoofguard* to monitor websides. *Spoofguard* will identify spoofing sides and warn the user. Again, we think, that this idea does not avoid phishing attacks, since the user may ignore the warning especially if there are two many false positives.

In 2006, there emerged a solution using visual artefacts to make it easier for an user to decide, whether a web-side is trustable or not [5]. The authors require an authentication of server and client using SSL. After having established the SSL connection, the user will see a so called "visual artefact" which only she and the server knows. As this solution requires client certificates, we run into the same problems as with SSL using mutual authentication, which is very inconvenient for the user.

In addition to bigger warning messages, there emerged the idea of producing better user passwords. The authors of [11] argue that most of the users use passwords, which are easy to crack. Furthermore, most of the users use the same password for every account. If a phisher was able to phish one, she may use it everywhere. Therefore, the authors of [11] proposed a tool, which calculates new passwords for every account, the user has. These passwords are then harder to crack then the ones, the user chooses herself. This approach does not avoid phishing, pharming or MITM attacks, but reduces their potential damage to one account.

In 2007, Oppliger et. al argued that the main problem of SSL connections is, that the SSL protocol is decoupled from the user authentication [9]. Such a setup makes MITM attacks easy. To overcome this problem, the authors propose to extend SSL in a way, that the original SSL connection and the user authentication are coupled. In our opinion, this approach does not avoid proxy MITM attacks, as the client may accidentally connect to a malicious but correctly certified server.

In [4], the authors propose the use of a so called "secure wallet" to store the user's credentials and to decide whether a webside is authentic or not. To protect also against malware phishing, one may use a protected execution environment. The authors want to use trusted computing support to realize their idea. To detect phishing servers, the secure wallet will store a fingerprint of the correct server, when it connects for the first time. If the fingerprint does not match anymore, the wallet will not release the credentials. Such a scheme is highly vulnerable to attacks on the first connect.

Another approach using a trusted device is proposed in [10]. The authors want to use such a device as second authenticator and to authenticate the server in a secure way. Their trusted device will not be built into a special computer, which allows the user to use every computer she wants. According to the authors, such a device may be a cellphone, a PDA or a smart watch. We think that always carrying such a device is not very convenient.

4. The TPM

The TPM as specified by the Trusted Computing Group (TCG) [14], is a module, which provides cryptographic functions and is able to store hashes securely in so called Platform Configuration Registers (PCRs). The TPM should be a low cost device, otherwise nobody will buy it. This leads to the need of only a minimal subset of cryptographic methods provided by the module and minimal hardware requirements.

The TPM is especially designed to act as root of trust for the collection and reporting of integrity metrics [14] [12]. The idea behind this concept is that a computer can be checked for integrity before usage. The TPM is the instance which stores all the validation data in a secure way to avoid an attack on this validation process.

The module holds confidential information like cryptographic keys in a protected environment. Furthermore, the TPM provides methods to “seal” sensible data to a special state of the platform. This means that these data may only be released if the platform is in a specified trustable state. A very important feature of the TPM is that this module can be uniquely identified. It is equipped with a so called Endorsement Key Pair, which is unique. The private part of this key pair is never released from the TPM which ensures the identification.

There are already several application for the TPM under several operating system. For Linux systems, there is the trusted bootloader called trustedGRUB [16] and a kernel patch to measure running applications [13]. For Windows (Vista) systems, there is a new feature called BitLocker used for drive encryption, which makes use of the TPM [8]. Furthermore, Windows (Vista) provides a secure Network Access Control using these modules [15].

5. Application Integrity

In this section, we will provide a mechanism to ensure application integrity using the TPM very similar to the one proposed in [7] and [13]. Our approach relies on the AEGIS approach¹ proposed by Arbaugh et. al. [1] in 1997. We extend the AEGIS approach by application checking and secure the mechanism using the TPM.

For the remainder of this section, we will first introduce secure bootstrapping and then continue with the validation of applications integrity.

5.1 Secure Bootstrapping

In 1997, Arbaugh et. al. proposed a new scheme called AEGIS for integrity checks² on boot-up [1], which was extended by themselves by an automated recovery protocol in case of a failure [17].

¹Originates from the classical mythology. Aegis is the shield or breastplate of Zeus or Athena.

²With integrity checks we mean calculating the hash value of a component and comparing this value with an older value of this component.

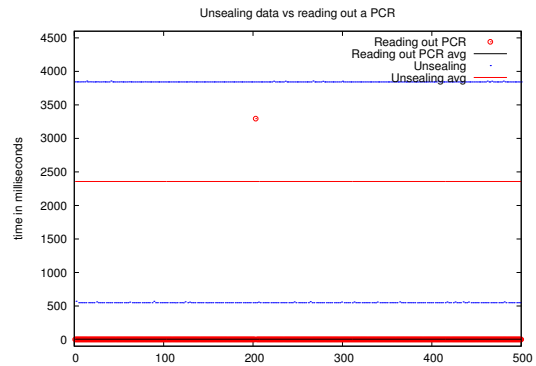


Figure 1. Comparison of unsealing data and reading out a PCR

In [1], the authors describe a secure bootstrapping protocol, where they establish a chain of trust. The chain starts with a small set of code, which is assumed to be valid. This first little program checks the Basic Input/ Output System (BIOS) for integrity. Afterwards, the BIOS validates the operating system kernel. If the hashes are valid, the operating system gets booted. Otherwise, a recovery kernel will be booted, which connects to a so called “Trusted Network Recovery Host”, which contains valid images of the desired code.

The idea of making conventional phishing, pharming and MITM attacks useless will apply to private users, which usually are not connected to a well configured network. Furthermore private users often administrate their computers by themselves. Therefore, the recovery procedure as described in the AEGIS approach [17] cannot be applied anymore. In our approach, the boot process will stop when invalid integrity values are detected and the user has to insert a recovery CD ROM to repair her system. If the system has been repaired successfully, a reboot is needed to trigger the validation checks again.

In the next section, we will present a scheme to check applications, if the operating system is running properly.

5.2 Measuring the Applications

In our approach there is the additional need to validate and recover applications. The number of applications, which may be used on private computers may be rather high, but the TPM contains only 16 free usable Platform Configuration Registers (PCRs). One solution for this problem is to concatenate all the hashes of the programs and calculate a new hash over the result as proposed by the TCG Main Specification [14]. This mechanism reduces the need for memory, since we need only one PCR per result. But this approach has one important disadvantage: it allows the operating system’s validation agent only to decide whether all applications are in a trustable state or not, but it does not help to decide which application is compromised if the test fails. To solve this problem there will be a file on the

harddisk, sealed to the bootstrapping integrity, which contains the individual hashes of the applications. This file will be used to identify the compromised application. Due to performance reasons, it is not desired to use this file to determine, whether all application are in a trustable state since reading out a PCR is significantly faster (factor 2000 as can be seen in Figure 1).

For the recovery, we again stop the boot process and require the user to insert a rescue CD ROM to repair its compromised application(s). Again if the application has been repaired successfully, the system needs to be rebooted to trigger to integrity checks again.

6. Strong Authentication using the TPM

If we have an integrity-checked e-commerce application running for which we know that it runs securely, we are basically done. So how can we ensure the untampered application's secure operation. As we focus on attacks stealing user credentials in this paper, our answer is *strong authentication*: if we can ensure that user credentials on their own do not suffice to get access to an e-commerce account and that what is needed in addition cannot be stolen via a social engineering attack (e.g. the phishing), then our system is secured against these types of attacks.

First of all, the concepts we are going to present in this section require mutual authentication by the merchant and by the client. Otherwise the presented approach would still be vulnerable to man-in-the-middle attacks. To achieve mutual authentication, we require a registration process between e-commerce provider and client, in which the merchant's and the client's public key are exchanged. This can be done by the merchant sending a CD ROM by registered mail containing the following:

1. the e-commerce' provider's public key with a piece of software sealing the key in the TPM of the clients machine,
2. a piece of software for client key generation and printing the client's public key's fingerprint³, sealing the client's private key in te TPM, and sending the client's public key to the e-commerce provider,
3. the e-commerce software using user credentials *and* a mutually authenticated challenge-response protocol where verification of the merchant's authentication information and computation of the client's authentication information is done by the TPM.

6.1 Authentication when Using the Registered Home Machine

Using e-commerce server/user key registration and sealing the keys in the TPM enables one to enforce strong authentication while the user is using his/her registered machine.

³The printed fingerprint will be sent to the merchant by registered mail for key verification.

The authentication process will work as depicted in Figure 2.

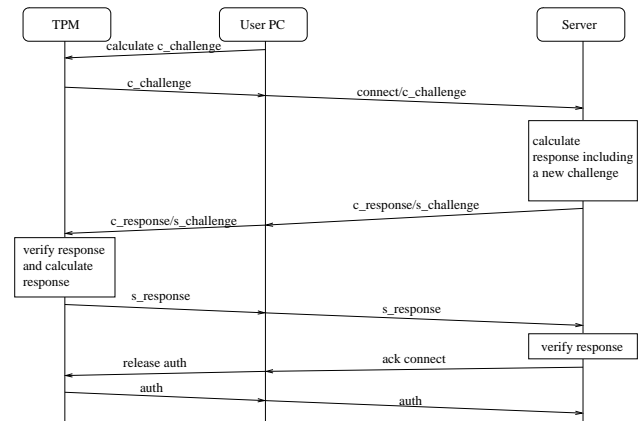


Figure 2. The authentication process from the user's registered home machine.

The authentication steps are the following:

1. the TPM has to calculate `c_challenge`, which will be used to authenticate the e-commerce server.
2. afterwards, the user tries to connect to the e-commerce server and sends `c_challenge`.
3. the server authenticates itself by calculating the appropriate response `c_response`. Furthermore, the server sends a new challenge `s_challenge` to the client to authenticate her.
4. using the TPM, the application on the user's machine verifies the server's response `c_response` and calculates a new response `s_response` for `s_challenge`.
5. the application on the user's computer asks the TPM to release the user credentials, which was sealed to a successful server authentication.
6. the application sends the authentication information to the bank.

(These steps do not mention time stamping and additional encryption of messages as they are not the core of the concept presented in this paper. They are obviously needed for the prevention of replay and man-in-the-middle attacks.)

6.2 Authentication when Using a Remote Machine and a Registered Mobile Device

Even though most users use most of the time their "home machine", they will infrequently access their on-line account from remote, unknown and therefore unregistered machines. If the user is in possession of a mobile device, and the e-commerce application can communicate with the

mobile device for instance via MMS or WLAN, then the user can also register his/her mobile device with the bank. Furthermore, the mobile device must have a private and a public key, where the latter is known by the e-commerce server. Then the mobile device can act on behalf of the home machine's TPM as depicted in Figure 3.

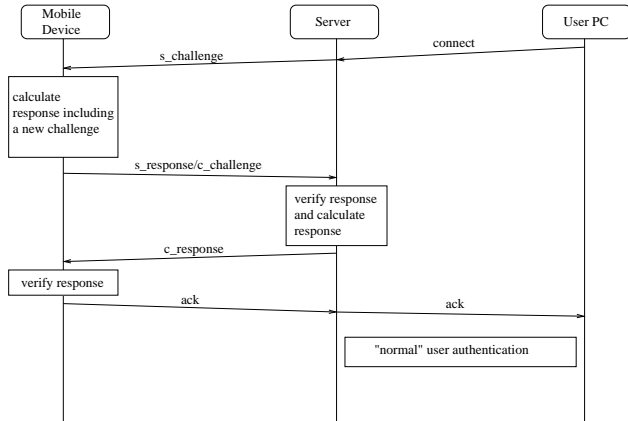


Figure 3. The authentication process when using a remote machine to a registered mobile device.

In that scenario, the authentication steps are as follows:

1. the user tries to connect to the e-commerce server,
2. which then sends an encrypted challenge `s_challenge` to the user's mobile device, where the user is asked confirm it.
3. the mobile device will then send back a response `s_response` including a new challenge `c_challenge` to authenticate the server.
4. if the mobile device was able to verify the server it sends an acknowledgment to it, which then acknowledges the connection request from the unknown computer.
5. in the end, the user has to authenticate herself to the e-commerce' user interface using her "normal" user credentials.

(As before, these steps do not mention time stamping and additional encryption of messages.)

As the server communicates directly with the mobile device using encrypted messages, this solution does not enable a MITM. But as the connection between the user's computer and the server is not really protected, this one is vulnerable to MITM. To reduce the risk, we require the user first to confirm the authentication process on the mobile phone and later all transactions. This way, there might be a man-in-the-middle, but he cannot cause harm as the user had to confirm it. In case, there is no user confirmation, an attacker could use formerly phished user credentials on an unknown computer and the mobile phone would

acknowledge silently everything. Furthermore, the attacker would be able to act as MITM and cause harm on the user's account.

6.3 Authentication with Unregistered devices

As not all users will possess a mobile device, which can store keys and execute additional programs, bypassing solutions must be available. We focus here on a solution which still requires a mobile phone, which is able to receive SMS (but nothing else), for ensuring strong authentication. The strong authentication relies on the authentication in mobile networks (e.g. GSM). The last solution is "clumsier" than the first two, but by being a little more complex ensures that phishing attacks still will not be possible. The basic idea is depicted in Figure 4.

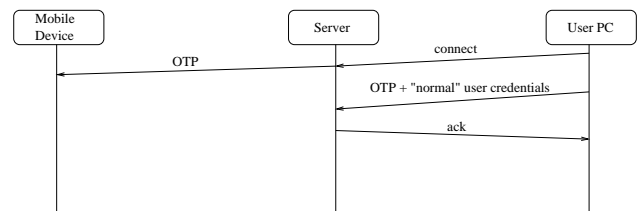


Figure 4. The authentication process when using only not-registered devices.

The authentication steps are as follows (it is assumed that the mobile phone number of the user was registered securely with the e-commerce provider):

1. the users tries to connect to e-commerce server,
2. which then an one time password (OTP) to the user's mobile phone.
3. the OTP has to send to the server using the unknown machine in addition to the user's "normal" credentials

(As before, these steps do not mention time stamping and additional encryption of messages.)

As this mechanism does not really provide prevention against proxy MITM attacks, the server has to send another OTP to the user's mobile phone, if a transaction will be done. The user then has to acknowledge this transaction with the OTP. Using such a mechanism, the user will be notified about every transaction on her account, which renders proxy MITM useless.

7. Conclusion

We have discussed a TPM (trusted platform module) based strong authentication process for access to remote application. As the assumption is too strong that e-commerce users always access the e-commerce application from their home machine, we presented in addition strong-authentication mechanisms for e-commerce applications which involve a

mobile phone (either registered or unregistered). In the case where the mobile phone is able to connect to a WLAN or receive MMS (which is reliable in contrast to SMS) and to execute additional applications, authentication can be performed very similar to the home-machine scenario. If the mobile phone may only receive SMS, we presented an approach in which SMSs containing one-time passwords can ensure strong authentication. Therefore, our approach can be seen somehow as a combination of [4] and [10].

We assume that everyone agrees that strong authentication is desirable in an e-commerce context and that the solution using the TPM on the home machine will not be questioned strongly. The mobile-phone bypassing-mechanisms are probably more debatable. However, today virtually everyone possesses a mobile phone, and instead of handing another different device to a user, using the device he/she carries with him/her anyway seems to be a desirable solution.

Finally, we did not only present an authentication approach, but also discussed a means of checking whether or not someone has tampered with the e-commerce application. Using the TPM helps us conducting integrity tests on user applications such as the e-commerce application. Hence the strong authentication cannot be bypassed by a modified application as it would not pass the integrity test. We believe strongly that e-commerce could benefit from our approach significantly as stealing user credentials will not give access to a user's e-commerce account anymore, and phishing and pharming attacks would become a threat of the past.

References

- [1] W. A. Arbaugh, D. J. Farber, and J. M. Smith. A Secure and Reliable Bootstrap Architecture. *IEEE Security and Privacy Conference*, pages 65 – 71, May 1997.
- [2] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J. C. Mitchell. Client-side defense against web-based identity theft. In *11th Annual Network and Distributed System Security Symposium (NDSS '04)*, San Diego, Feb. 2004.
- [3] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 581–590, New York, NY, USA, 2006. ACM Press.
- [4] S. Gajek, A.-R. Sadeghi, C. Stble, and M. Winandy. Towards multicolored computing - compartmented security to prevent phishing attacks. In *Workshop on Information and System Security (WISSEC'06)*, Antwerpen (Belgium), 2006.
- [5] S. Gajek, J. Schwenk, and C. Wegener. Ssl-va-authentifizierung als schutz von phishing und pharming. In *Sicherheit*, pages 6–17, 2006.
- [6] A. Herzberg and A. Gbara. Protecting (even) naive web users, or: preventing spoofing and establishing credentials of web sites, 2006.
- [7] C. Latze and U. Ultes-Nitsche. Using a trusted platform module to enable a secure usage of nodes in company networks - an extension of the aegis approach. In *Proceedings of the KiVS 2007 Workshop on Secure Network Configuration (KiVS NetSec 2007)*, pages 13–20, 2007.
- [8] Microsoft. Bitlocker drive encryption: Executive overview, 2006. available at <http://technet.microsoft.com/en-us/windowsvista/aa906018.aspx>, last visited July 2007.
- [9] R. Oppliger, R. Hauser, D. Basin, A. Rodenhaeuser, and B. Kaiser. A proof of concept implementation of ssl/tls session aware user authentication (tls-sa). In T. Braun, G. Carle, and B. Stiller, editors, *Kommunikation in Verteilten Systemen*, Informatik Aktuell, pages 225–236. Springer Verlag, 2007.
- [10] B. Parno, C. Kuo, and A. Perrig. Phoolproof phishing prevention. In *Financial Cryptography*, 2006.
- [11] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. C. Mitchell. Stronger password authentication using browser extensions. In *Proceedings of the 14th Usenix Security Symposium*, 2005.
- [12] S. Pearson, B. Balacheff, L. Chen, D. Plaquin, and G. Proudler. *Trusted Computing Platforms. TCPA Technology in Context*. Prentice Hall, 2003.
- [13] R. Sailer, X. Zhang, T. Jaeger, and L. van Doorn. Design and implementation of a tcb-based integrity measurement architecture. In *Proceedings of the 13th USENIX Security Symposium*, 2004.
- [14] The Trusted Computing Group. *Trusted Computing Platform Alliance. Main Specification Version 1.1b*. Trusted Computing Group, 2003. available at <https://www.trustedcomputinggroup.org/specs/TPM>.
- [15] Trusted Computing Group and Microsoft. Standardizing network access control: Tnc and microsoft nap to interoperate, May 2007.
- [16] University of Bochum. TrustedGRUB. available at <http://sourceforge.net/projects/trustedgrub>, last visited July 2007.
- [17] W. A. Arbaugh, A. D. Keromytis, D. J. Farber, and J. M. Smith. Automated Recovery in a Secure Bootstrap Process. *Internet Society 1998 Symposium on Network and Distributed System Security*, pages 155 – 167, 1998.