E-COMMERCE AUTHENTICATION
An Effective Countermeasures Design Model

Victor D. Sawma, Robert L. Probert
School of Information Technology and Engineering, University of Ottawa, 800 King Edward, Ottawa, Ontario, Canada
Email: vsawma@site.uottawa.ca, bob@site.uottawa.ca

Keywords: requirements analysis, electronic commerce, security, authentication, Secure Electronic Transactions (SET)

Abstract: Existing authentication models for e-commerce systems take into account satisfying *legitimate user requirements* described in security standards. Yet, the process of introducing countermeasures to block *malicious user requirements* is ad hoc and relies completely on the security designer expertise. This leads to expensive implementation life cycles if defects related to the design model were discovered during the system-testing phase. In this paper, we describe an authentication countermeasures design model for e-commerce systems. This model includes effective countermeasures against all known malicious user requirements and attacks. The described model is preventive in nature and can be used with other authentication models or can be implemented as a stand-alone module for e-commerce systems.

1 INTRODUCTION

The use of e-commerce (EC) has grown exponentially in recent years [Hobley, 2001] and this growth requires a comparable growth in security presence. The Common Criteria (CC) Redbook [CC, 1999] was one of the first attempts to standardize security assessment requirements for Information Technology (IT) systems. Security requirements, in this context, can be divided into two subsets. The first set includes *legitimate user requirements* and, thus, must be satisfied. The second set is the set of *malicious user requirements*. These requirements might allow malicious users to breach system security and must not be satisfied via proper security countermeasures. This requires EC systems, a subset of IT systems, to include proper countermeasures against all types of known security attacks in order to satisfy standard security evaluation requirements described in [CC, 1999].

At the time of writing this paper, the process of introducing countermeasures in an EC system design phase relied primarily on security designer expertise. Moreover, the countermeasure selection process is ad hoc and, thus, a prescribed countermeasure at system design time might prove inadequate during the system-testing phase. This might result in expensive rework for fixing defects related to the design model. [Treese, 1998]

In this paper, we will describe a countermeasures design model for authentication in e-commerce systems. This model was the result of applying a new methodology for deriving effective countermeasures design models for e-commerce systems (illustrated in Figure 1). For details, refer to [Probert, 2003]

The contribution of our design model is outlined as follows:

- It satisfies authentication security requirements and blocks malicious user requirements at system design time
- It is effective against all known security attacks related to e-commerce authentication
- It can be directly integrated into high-level design documents of e-commerce systems
- It can be used with security-aware technologies such as SET or can be implemented as a stand-alone module for EC systems

In addition, in this paper we provide an overview of all known security attacks related to e-commerce authentication.

The remainder of this paper is organized as follows. In Section 2, we define authentication along with a set of related security attacks. Section 3 describes each security attack, derives *attack enablers* (properties of a security feature which are useful for enabling a security attack), and prescribes proper countermeasures. Section 4 shows the derived countermeasures design model and discusses its effectiveness. Section 5 is a case study on a SET-

integrated e-commerce system. Section 6 concludes and provides a summary. Finally, we describe future work in Section 7.

Phase 1: Select features and derive design models

1. Select security features.
2. For each security feature, derive a countermeasures design model:
   - 2.1. Identify and abstract all attacks related to the security feature.
   - 2.2. For each security attack:
     - 2.2.1. Derive all attack enablers.
     - 2.2.2. For each attack enabler
       - 2.2.2.1. Prescribe appropriate *security countermeasures*.
       - 2.2.2.2. For each countermeasure:
         - 2.2.2.2.1. Analyze countermeasure for residual vulnerabilities.
         - 2.2.2.2.2. Add corrective measures to overcome vulnerabilities.
   - 2.3. Derive the complete security-oriented countermeasures design model.
     - 2.3.1. Group all prescribed countermeasures
     - 2.3.2. Separate countermeasures into action countermeasures and underlying countermeasures.
     - 2.3.3. Divide countermeasures design model into an action-box and underlying planes.
       - 2.3.3.1. Put action countermeasures inside box [group into flowcharts]
       - 2.3.3.2. Add underlying countermeasures
   - 2.4. Verify attack coverage via traceability matrix.

Phase 2: Instantiate and Integrate the derived models into an e-commerce system design

3. Instantiate the Countermeasures Design Models for Each Security Feature
4. Integrate the Instantiated Models into an Existing Design

Figure 1. Authentication security attacks, attack enablers, and countermeasures.

2 AUTHENTICATION AND RELATED SECURITY ATTACKS

Authentication is the process of verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in a system. [NIST, 2001] The identity of a certain user or process is challenged by the system and proper steps must be taken to prove the claimed identity.

While lots of research described authentication models that satisfy legitimate user requirements, little has been done, at the system design level, to prevent malicious user requirements from occurring. In this paper, we will address this gap by describing a countermeasures design model that incorporates all proper countermeasures into the high-level system design *during the design phase*.

Security attacks related to authentication were identified from the literature and from personal experience. In this research, we have projected the identified security attacks onto different types of EC authentication models described in [Wilson, 1997], [Ford, 1997], [Agnew, 2000], [Bellare, 1998], [Hawkins, 2000] and [Chang, 2002]. The result was a set of six attacks applicable to authentication in the domain of EC systems. These attacks are:

1. Sniffing attacks
2. ID spoofing attacks
3. Brute-force attacks
4. Dictionary attacks
5. Credential decryption attacks
6. Replay attacks

It is important to note that the main focus of this paper is on attacks directly related to e-commerce systems. Attacks related to network components, third-party software components, and attacks against the operating system that is supporting the e-commerce system will not be discussed.

3 ATTACK ENABLERS AND COUNTERMEASURES

This section provides a description of the above authentication-related security attacks. For each attack, enablers are then derived, and proper countermeasures are prescribed.

3.1 Sniffing attacks

Sniffing attacks ([Herzog, 2001], [Viega, 2002], [Nguyen, 2001], and [Schneier, 2000]) (also known as the man-in-the-middle attacks) are the digital analogues to phone tapping or eavesdropping. This

type of attacks captures information as it flows between a client and a server. Usually, a malicious user attempts to capture TCP/IP transmissions, because they may contain information such as usernames and passwords. A sniffing attack is often classified as a man-in-the-middle attack because in order to capture packets from a user, the machine capturing packets must lie in between the two systems that are communicating. The *attack enabler* in this case is the process of sending data across communication channels in clear text format.

Preventing access to the communication channel is not a valid countermeasure in this case due to the open nature of the Internet. By encrypting the communication channel between the user/process and the system, sniffing attacks can be defeated, i.e., sniffing cannot retrieve any useful information.

3.2 ID spoofing attacks

ID spoofing attacks ([Herzog, 2001], [Viega, 2002], [Nguyen, 2001], [Schneier, 2000], and [Anderson, 2001]) occur when a malicious user or process claims to be a different user or process. This attack allows an intruder on the Internet to effectively impersonate a local system's IP address. If other local systems perform session authentication based on the IP address of a connection, they will believe incoming connections from the intruder actually originate from a local "trusted host" and will not require a password. The *attack enabler* is when authentication relies on static information such as IP addresses, host names, etc. This means trusting certain hosts or processes through some pre-defined static information. The system will authenticate the user or process by checking the given static information. In such a case, the attacker will attempt, through complex attack tools, to "spoof" the system by claiming that he is the trusted host or process. Since no challenge is required in this case, the attack has a great chance of succeeding. The *countermeasure* for this attack is to use challenge-based authentication which includes the use of certificates, user/password combinations, etc.

If challenge-based authentication is inapplicable for a certain specific case, then *least privilege* static authentication must be taken into consideration. Least privilege static authentication means giving the least possible access privilege to the least possible number of users, processes or hosts after successful authentication.

3.3 Brute-force attacks

A Brute-force attack ([Herzog, 2001], [Viega, 2002], and [Anderson, 2001]) is any form of attack against a credential information file that attempts to find a valid username and password in succession. This type of attack is *enabled* by gaining access to the credentials' (user names and passwords) storage medium. The attacker retrieves a copy of the database system or system file preserving credential information. If the credential information is encrypted, a brute-force attack tool will try all possible combinations of user names and passwords. For each combination, the user name and password

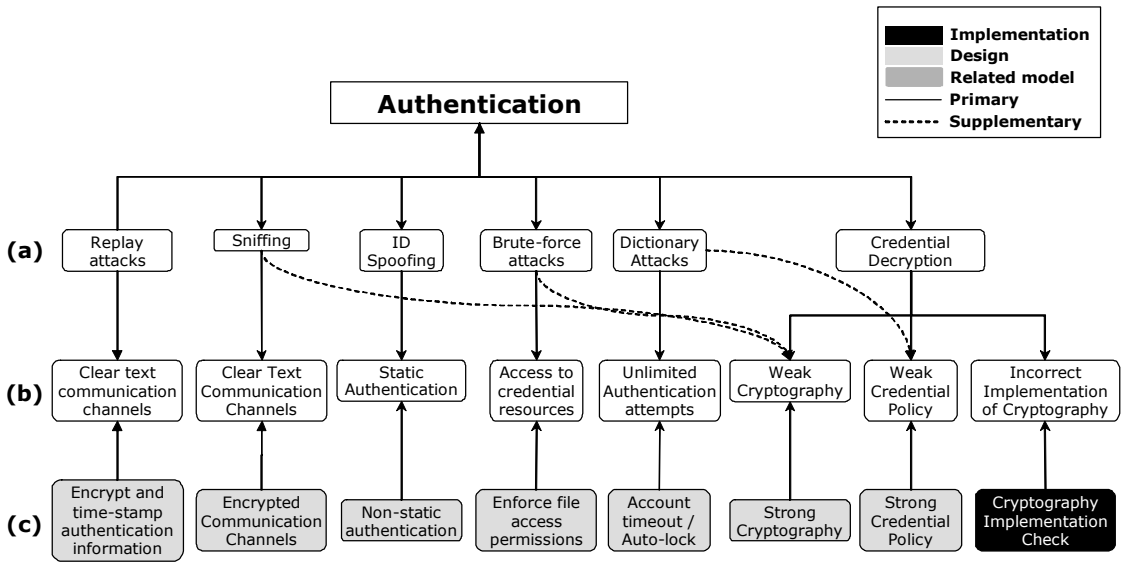

Figure 2. (a) Authentication security attacks, (b) attack enablers, and (c) countermeasures.

are encrypted using the same encryption algorithm that was originally used to encrypt the credential information. Then, the encrypted credential data are compared to the retrieved copy of original credential data. Different types of encryption algorithms are used and the attack proceeds until both credentials (user name and password) match. The *countermeasure* for this type of attacks is to enforce access permissions through a strong access control policy at the operating system level.

3.4 Dictionary attacks

A *dictionary attack* ([Viega, 2002], and [Anderson, 2001]) is the "smart" version of brute-force attacks and is also executed using automated attack tools. Yet, these tools are capable of working on web interfaces without access to credential information storage mediums. These tools only require the prior knowledge of a valid system user name. Once given a user name, the attack tool will try all possible combinations of that user name with a huge database (such as a dictionary) of possible passwords. This attack has a high probability of succeeding since we, as humans, tend to use passwords that are easy to remember. The *attack enabler* is a "high" number of allowed consecutive unsuccessful authentication attempts. The *countermeasure*, in this case, is to prevent the attack automation by setting an upper limit to the allowed number of unsuccessful authentication attempts. This can be done through an account auto-lock or timeout procedure. In other words, when a certain number of consecutive unsuccessful authentication attempts is reached, the system will automatically lock or disable the account and will alarm the system administrator. Enabling or unlocking the account can either be done by the user or can be done automatically by the system after a certain period of time.

A residual vulnerability for account auto-locks or timeouts is when malicious users target them as means for denial of service attacks ([Herzog, 2001], [Viega, 2002], [Nguyen, 2001], and [Treese, 1998]). The *residual vulnerability* is when these countermeasures prevent legitimate users from using the EC system because their account has been disabled. This contradicts with an essential security objective: availability [NIST, 2001] The *countermeasure* for this residual vulnerability is to allow legitimate users to unlock their account through an easy-to-perform process that can be done at any time. A discussion of a similar case is provided later in our case study.

3.5 Replay attacks

A *replay attack* means that the malicious user trapped the authentication sequence that was transmitted by an authorized user through the network, and then replayed the same sequence to the server to get himself authenticated. [Anderson, 2001] The *attack enabler* in this case is, again, access to the communication channel and data sent in clear text format. The proper countermeasure is to encrypt and time-stamp all sensitive data sent across the communication channel. By doing so, this type of attacks can be defeated.

3.6 Credential decryption attacks

Credential decryption is a basic supplementary attack for sniffing attacks, brute-force attacks, and dictionary attacks. A tool, whose aim is to break the encryption algorithm that was used to encrypt credential information, usually performs these attacks. [SecuriTeam] A*ttack enablers* for this attack might be a weak credential policy, a weak cryptographic algorithm, or an incorrect implementation of the cryptographic algorithm. A *weak credential policy* increases the probability of a dictionary attack's success and its *countermeasure* is to have a strong credential policy. *Weak cryptography*, on the other hand, increases the probability of a brute-force attack or a sniffing attack to succeed and its countermeasure is to use a strong cryptographic algorithm. The countermeasure to an *incorrect implementation of cryptography* is to check the cryptographic algorithm after implementation and, thus, cannot be done at design time.

Although not included at system design time, "checking the cryptographic algorithm after implementation" is now a system requirement. In other words, the system design specification must satisfy this countermeasure and provide guidelines for implementation.

3.7 Side-Channel Attacks

In cryptographic devices such as smart cards, data other than input data and output data may 'leak out' during cryptographic procedures. Computation timing is one kind and so is power consumption. This is simply because the smart card uses an external power source. [Kocher] developed the side channel attack in which an attacker infers stored secret information in a cryptographic device by using such leaked data. This type of attack, which includes timing attack, Simple Power Analysis

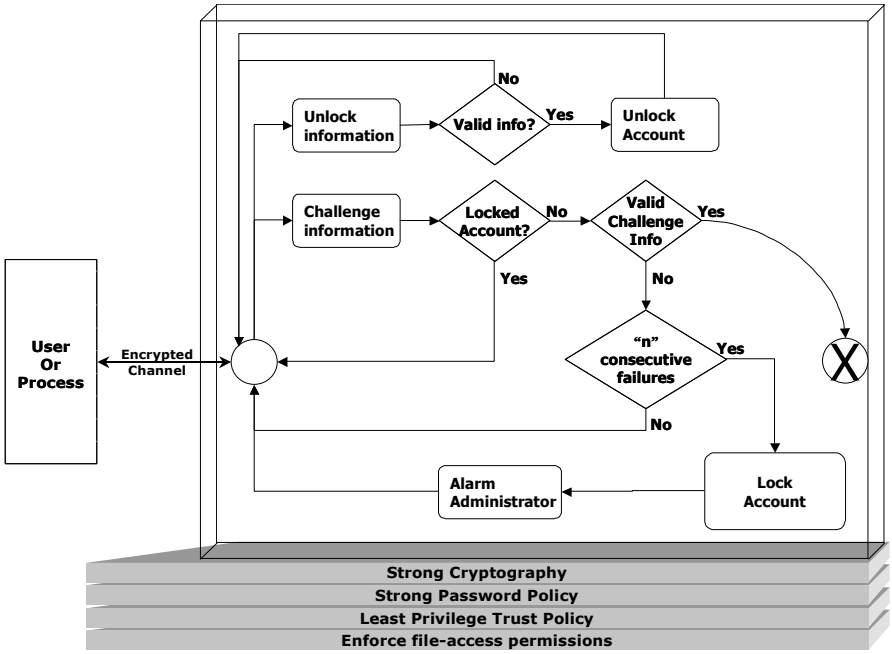

Figure 3. The e-commerce authentication countermeasures design model

(SPA) attack, and differential power analysis (DPA) attack, render smart cards particularly vulnerable.

For the purpose of our discussion, smart cards might be used for authentication purposes but only at the client side. In our case, the card and the external power source are both assumed to be secure since the system client is responsible for protecting them. The main emphasis of our research is on securing the EC system itself; accordingly, we do not treat this type of attack in this paper. Yet, it is important that security designers be aware of the existence of this type of attacks.

3.8 Summary

Figure 2 shows a summary of security attacks related to authentication along with the attack enablers and prescribed countermeasures.

Access to credential resources and *weak cryptography* are two attack enablers for brute-force attacks. The first provides access to the medium in order to retrieve credential information and the second allows for a low-cost security attack.

Weak credential policy, *weak cryptography*, and *incorrect implementation of cryptography* are three attack enablers for *credential decryption* attacks. A weak credential policy allows system users to select easy-to-guess passwords. Weak cryptography, on the other hand, allows for a low-cost security attack.

An incorrect implementation of the cryptographic algorithm can be seen as an implementation defect and can only be checked after the system is implemented. Yet, it is incorporated into the system design as a security requirement.

Weak cryptography is a supplementary attack enabler for sniffing attacks. *Sniffing* and *replay* attacks both rely on a clear text communication channel. [Herzog, 2001]

4 AUTHENTICATION COUNTERMEASURES DESIGN MODEL

After identifying the attacks, attack enablers, and countermeasures for authentication, the prescribed countermeasures are grouped and ordered in a countermeasure design model.

Figure 3 shows the countermeasures design model derived by our methodology [Probert, 2003] for e-commerce authentication. This model is detailed enough to be incorporated into high-level design documents of EC systems. Furthermore, a faithful implementation of the model will lead to an e-commerce system that is resistant to all known authentication security attacks.

The validation trace-ability matrix in Figure 4

COUNTERMEASURES	AUTHENTICATION SECURITY ATTACK TYPE						
		Sniffing	ID Spoof.	Brute-force	Dictionary	Replay	Cred. Dec.
	Encrypted comm. Channels	D				D	
	Non-static auth.		D				
	Enforce file-access permissions			D			
	Account timeout / Auto-lock				D		
	Strong crypt.						D
	Strong credential policy						D
	Cryptography impl. Check						I

D = design countermeasure, I = implementation countermeasure

Figure 4. Effectiveness of the authentication countermeasures design model.

shows the security coverage and effectiveness of our authentication countermeasures design model in relation to the relevant authentication security attacks. A "D" implies a countermeasure is related to the system design. An "I" implies a countermeasure is related to the system implementation. Yet, in both cases, the countermeasure is introduced at system design time.

Sniffing attacks, ID spoofing attacks, brute-force attacks, dictionary attacks, weak cryptography, and weak credential policy attacks required design countermeasures. The only countermeasure to be introduced after system implementation is "cryptography implementation check."

5 CASE STUDY: APPLYING THE DERIVED AUTHENTICATION COUNTERMEASURES DESIGN MODEL

In this section, we will apply the derived authentication countermeasures design model to an e-commerce system application. The system we describe is a SET-integrated e-commerce system that allows its users to select products from a catalog, place orders, submit orders, and pay online.

Secure Electronic Transactions (SET) is an open technical standard for the commerce industry developed by Visa and MasterCard, in conjunction with leading computer vendors such as IBM, as a way to facilitate secure payment card transactions over the Internet. SET is an open standard for protecting the privacy, and ensuring the authenticity, of electronic transactions. [SET, 1997]

Our e-commerce system, whose collaboration diagram is shown in Figure 5, is a virtual store system that involves four parties: merchant EC system, client, system administrator, and payment gateway. In this case study, we will apply our countermeasures design model to the merchant e-commerce system only. Every party requiring a SET implementation will have its own SET module. This SET module is assumed to be compliant to the SET specification of that entity.

5.1 Need for additional security

SET secures the transaction while in progress, but does not take into consideration the security of data on cardholder, merchant, and payment gateway systems including protection from viruses, trojan horse programs, and hackers. [SET, 1997]

Additional security is also required to address possible SET conflicts. [Treese, 1998] This includes, but is not limited to the following:

- Protecting card numbers from malicious use when card numbers are revealed to the merchant
- Ensure proper client authentication since cardholder certificates are optional
- Secure the transmission of transaction order information
- Provide non-repudiation

The conflicts discussed above suggest that SET security is indirectly dependent on the EC system security. For example, if a malicious user successfully penetrates the SET-integrated e-commerce system by breaching the system

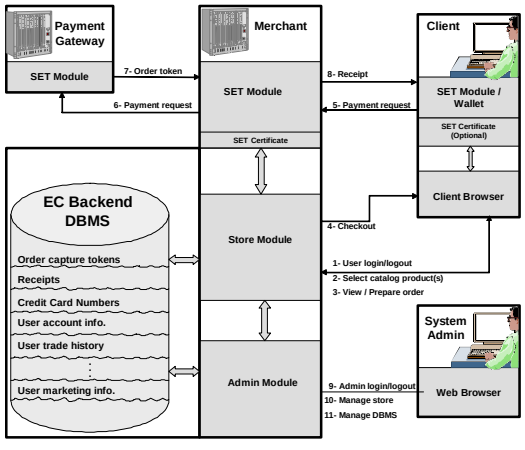

Figure 5. Our SET-integrated e-commerce system.

authentication, this user might be able to capture credit card numbers of legitimate system users provided through SET. This contradicts an important security objective for both SET and the e-commerce system: confidentiality [NIST, 2001].

5.2 Applying authentication

In order to apply our countermeasures design model, we must instantiate its features. This means that every feature of the model must be specified for system implementation.

A description of how we instantiated each feature of our authentication countermeasures design model for the SET-integrated e-commerce system is provided below and shown in Figure 6.

5.2.1 Encrypted channel

The "encrypted channel" feature is instantiated to SSL (Secure Sockets Layer) [Freier, 1996]. SSL is a protocol developed by Netscape Communications Corporation to provide security and privacy over the Internet. This protocol supports server and client authentication, is application independent, and allows HTTP (HyperText Transfer Protocol) to be layered on top of it transparently. Furthermore, SSL is optimized for HTTP and is able to negotiate encryption keys as well as authenticate the server before the web browser exchanges data. The SSL protocol maintains the security and integrity of the transmission channel by using encryption, authentication and message authentication codes. In our case, SSL is selected because it provides an encrypted channel of communication with no requirements at the client side. The only client requirement is to have an SSL enabled web browser. The majority of existing web browsers, and all major web browsers such as Internet Explorer™, Netscape Communicator™, and Mozilla™, have built-in SSL support.

5.2.2 Challenge information

The "challenge information" feature is instantiated to become "user name and password". The reason for our selection is to avoid client setup complications inherited by requiring system users to have a client certificate or a smart card. A user name and a password with a strong password policy are enough to provide the required security presence from this feature.

5.2.3 Consecutive failures

The number of allowed consecutive failures before locking a client user account is instantiated to 10 (ten) consecutive attempts. The reason for this selection is to have a number that is enough for legitimate users to fix normal mistakes such as typing mistakes or having the keyboard CAPS LOCK key on. On the other hand, this number is also appropriate for preventing malicious users from performing dictionary attacks.

5.2.4 Unlocking an account

In order to unlock a locked account, the e-commerce system will send an e-mail message to the system client user external mailbox once the account is locked. This message will contain a URL with a randomly generated hashed string related to the user account. The system user will have to visit the provided URL either by clicking on the link or by a copy/paste action into the web browser. The URL will point to a program script at the e-commerce system side. The system will then check if the provided link contains valid information for both the user account and the hashed string. If so, the account is unlocked and the user will be allowed to log into the system again. Otherwise, the system will keep the account locked and the system administrator is alarmed.

The reason for having this process to unlock an account is because legitimate users can perform it at any time. Other unlocking processes, such as calling customer service, might contradict with system availability. An example is when a client user does not call within customer service hours. In this case, a legitimate user might be blocked from using the system until a customer service representative is available to unlock his account and, thus, the system availability requirement is violated.

One can argue that sending an e-mail message to unlock the account would rely on the security of the user mailbox. In other words, a malicious user might be able to unlock a locked account by breaking the e-mail system security. In this case, the worst case is that the malicious user will be able to have another 10 (ten) attempts before the system locks the account again and alarm the system administrator. Even in this worst-case scenario, the risk of having a malicious user successfully breaking into the system is very low. Furthermore, by having a strong password policy, a lock/unlock feature, and an administrator alarm, breaching system authentication in this case is very hard and almost impossible.

5.2.5 Strong cryptography

The "strong cryptography" feature is instantiated to use the RSA cryptosystem with 1024 bits keys. The reason for selecting RSA is because it is a standard for secure cryptography. Furthermore, several recent

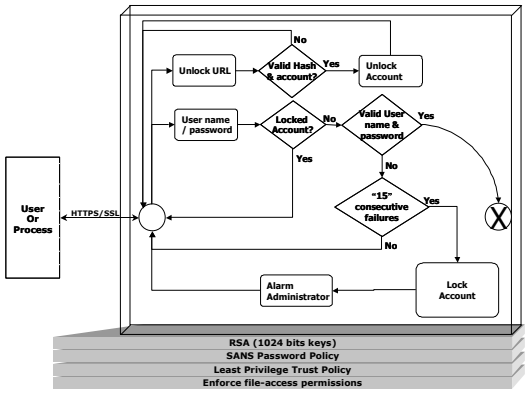

Figure 6. The SET-integrated system authentication countermeasure design model.

standards specify a 1024-bit minimum for corporate use. Less valuable information may well be encrypted using a 768-bit key; as such a key is still beyond the reach of all known key-breaking algorithms. [Rivest, 1978]

5.2.6 Strong password policy

The "strong password policy" feature is instantiated to obey the SANS standard [SANS] for strong password policies. This includes but is not limited to:

- Changing all system-level passwords (e.g., root, NT admin, application administration accounts, etc.) at least on a quarterly basis
- Changing all user-level passwords (e.g., email, web, desktop computer, etc.) at least every six months. The recommended change interval is every four months
- Passwords must contain both upper and lower case characters (e.g., a-z, A-Z)
- Passwords must have digits and punctuation characters as well as letters e.g., 0-9, !@#$%^&*()_+|~-=\`{}[]:";'<>?,./)
- Passwords must be at least eight alphanumeric characters long
- Passwords are not a word in any language, slang, dialect, jargon, etc.

6 SUMMARY

Satisfying security requirements is one of the most important goals for e-commerce system security designers. Yet, the process of introducing countermeasures to EC system design models is ad hoc in nature and relies primarily on the security designer expertise. Thus, a countermeasure that is prescribed at system design time might prove inadequate during the system-testing phase and might result in expensive fixing cycles. [Treese, 1998] In this paper, we described an authentication countermeasures design model for e-commerce systems. We also demonstrated its effectiveness and provided a case study by applying it to a SET-integrated e-commerce system.

As a result, we conclude that our authentication countermeasures design model and methodology for e-commerce systems:

- Capture and satisfy authentication security requirements at system design time
- Block malicious user requirements at system design time
- Can be directly incorporated into high-level design documents of e-commerce systems
- Contain effective countermeasures against the set of all known security attacks related to authentication
- Provide countermeasures in a chronological order. Thus, converting the model into a flow chart for implementation purposes is straightforward and can be easily done
- Provide guidelines for avoiding security pitfalls during system implementation
- Can be used with security-aware technologies such as CORBA™ or can be implemented as a stand-alone module for EC systems

Furthermore, in this paper we provided an overview of all known security attacks related to e-commerce authentication

7 FUTURE WORK

Our future work includes deriving countermeasures design models for other security properties and processes such as authorization, access control, transaction privacy, etc. The purpose of this research, in general, is to be able to provide complete e-commerce security design models that satisfy all legitimate user requirements and block all malicious user requirements. These models, at the same time, must satisfy enterprise security standards requirements.

REFERENCES

Agnew, G. 2000. "Cryptography, Data Security, and Applications to E-commerce." *Electronic*

Commerce Technology Trends Challenges and Opportunities. 69-85. IBM Press.

Anderson, R. 2001. "Security Engineering: A Guide to Building Dependable Distributed Systems", John Wiley & Sons. ISBN: 0-471-38922-6

Bellare, M., Canetti, R. and Krawczyk, H. 1998. "A Modular Approach to the Design and Analysis of Authentication and Key Exchange Protocols."

CC Common Criteria for Information Technology Security Evaluation (1999). Retrieved October 9, 2001 from http://commoncriteria.org/

Chang, K. Lee, B. and Kim, T. 2002. "Open Authentication Model Supporting Electronic Commerce in Distributed Computing", *Electronic Commerce Research*, Vol. 2, 135-149, Kluwer Academic Publishers

Ford W. and Baum, M. 1997 "Secure Electronic Commerce", Prentice Hall. ISBN 0-13-476342-4

Freier, A. Karlton, P., and Kocher, P. 1996 "The SSL Protocol Version 3.0". Retrieved from http://wp.netscape.com/eng/ssl3/draft302.txt

Hawkins, S., Yen, D., and Chou, D. 2000. "Awareness and Challenges of Internet Security", *Information Management & Computer Security*, Vol. 8(3), MCB University Press.

Herzog, P. 2001. "The Open Source Security Testing Methodology Manual", version 1.5. Retrieved from http://ideahamster.org/

Hobley, C. 2001. Just Numbers: Numbers on Internet use, electronic commerce, IT and related figures for the European Community. Retrieved from http://europa.eu.int/

Kocher, C. (n.d.) "Cryptanalysis of Diffie-Hellman, RSA, DSS, and Other Systems Using Timing Attacks" Retrieved from http://www.cryptography.com/

NIST, National Institute of Standards and Technology, 2001. "Underlying Technical Models for Information Technology Security", 2001. Special Publication 800-33. Retrieved from: http://csrc.nist.gov/publications/

Nguyen, H. 2001. "Testing Applications on the Web", 285-310, John Wiley & Sons.

SANS, "Password Policy", Retrieved from http://www.sans.org/newlook/resources/policies/

Probert, R., Sawma, V., 2003. E-Commerce Security: A New Methodology for Deriving Effective Countermeasures Design Models. To appear in the proceedings of the 16th Annual Federal Information Systems Security Educator's Association (FISSEA) conference. Website: http://csrc.nist.gov/organizations/fissea/

Rivest, R., Shamir, A. and Adleman, L. 1978. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, 21(2), 120-126.

Schneier, B. 2000. "Secrets and Lies: Digital Security in a Networked World", John Wiley & Sons

SecuriTeam Security Tools Archive. (n.d.) Retrieved from http://www.securiteam.com/

SET Secure Electronic Transaction Specification. 1997. Book 1: Business Description, Version 1.0. Retrieved from http://www.setco.org/

Treese, G. and Stewart, L. 1998. Designing Systems for Internet Commerce. Addison-Wesley.

Viega, J. and McGraw, G. 2002. "Building Secure Software", Addison-Wesley.

Wilson, S. 1997. "Certificates and trust in electronic commerce", *Information Management & Computer Security*, Vol. 5(5), 175-181, MCB University Press.