

Method and Apparatus for Protecting Data in Computers

Field of the Invention

The present invention relates to the field of protecting private data in computers using cryptographic techniques.

Background

In this age of pervasive computing, many end user devices, including desktop computers, laptop computers and various mobile devices, process and exchange large amount of information in wired as well as wireless environments. Some of this information is highly sensitive, such as private personal information and perpetrate corporation information. Information that can benefit a user or organization can also be used against the user or the organization if it falls into wrong hands. Industrial espionages among highly competitive businesses are resorting to electronic means to stealing corporate information.

Many information security and network security solutions have been developed. Firewalls are designed to shelter computers and computer networks from the ravages of the outside backers. Intrusion detection systems are designed to detect attacks to computers or networks. Virtual private networks are designed to allow secure corporate communications over the public Internet. While these solutions can prevent certain attacks to computers within a network, they are not designed to protect data in end user devices which are probably the most vulnerable points in the chain of information security.

Encryption is the most pervasive solution for providing data confidentiality. Most computer data encryption software products are based on symmetric key encryption algorithms (such as Data Encryption Standard and Advanced Encryption Standard) where the same secret key is used for both encryption and decryption operations. To use the software, a user must enter a correct password. A serious problem with this kind of software products is that they suffer from "dictionary attack". It is a well known problem in the computer security community that human selected passwords are inherently insecure since majority of the users choose passwords that are of low entropy (e.g., short and/or easy to remember passwords). Such passwords allow an attacker to perform dictionary attack, i.e., attempt to login to a data encryption software by trying all possible passwords until he finds the correct one.

Thus, there exists clearly a need for a data encryption method which is able to prevent the dictionary attack. In order for such a method to be acceptable to mass users, it must also be easy to use and efficient in operation.

SUMMARY

According to one aspect of the invention, there is provided a method and apparatus for encrypting and decrypting data in a host computer based on an external portable token and a user password, the method including the steps of:

System initialization whereby

- Installing Virtual Disk Manager software in the host computer;
- Generating a random Token Batch Key and storing the Token Batch Key in a plural number of external portable tokens;
- Promoting a user selecting a Name, a Password, a Counter and a random Token Key; said Name, Password, Counter and Token Key are stored in one of the external portable tokens, called the master token.

Backing up of external portable token

- Encrypting the Name, Password, Counter and Token Key in the master token using the Token Batch Key;
- Transferring the ciphertext to another external portable token in the same batch as the master token;
- Decrypting the ciphertext using the Token Batch Key to recovering Name, Password, Counter, and Token Key and storing them in the receiving external portable token.

Creating secure virtual disk on host computer

- Receiving user request to create a secure virtual disk on the host computer;
- Promoting the user to plug-in an external portable token and input his/her password;
- Asking user to specify name, size and location of the secure virtual disk;
- Creating the secure virtual disk based on user's specifications;

Accessing secure virtual disk for data encryption and decryption

- Receiving user request to access the secure virtual disk on the host computer;
- Promoting the user to plug-in an external portable token and input his/her password;
- Mapping the secure virtual disk into a virtual disk for user access;
- Encrypting files written into the virtual disk using the token key;
- Decrypting files read from the virtual disk using the token key.

Correspondingly, an apparatus and a computer program product based on the foregoing method are also disclosed.

Brief Description of the Drawings

Embodiments of the invention are described hereinafter with reference to the drawings, in which:

FIG. 1 is a block diagram illustrating a host computer which interfaces with an external portable token through a communication link.

FIG. 2 describes the configuration of the external portable token.

FIG. 3 shows the configuration of the re-writable nonvolatile memory in the external portable token.

FIG. 4 is a flowchart of the operation during system initialization.

FIG. 5 is a flow chart of the operation during creation of a secure virtual disk.

FIG. 6 is the flow chart of the operation during user access to a secure virtual disk.

FIG. 7 describes the data structure of an encrypted file.

FIG. 8 shows the flow chart of the operation during file encryption.

FIG. 9 shows the flow chart of the operation during the file decryption.

Detailed Description

A method, an apparatus and a computer program product for creating, accessing and closing a secure virtual disk in a host computer are described. In the following description, numerous details are set forth including encryption, decryption, file writing and file reading for examples. It will be apparent to one skilled in the art, however, that the present invention may be practiced without these specific details. In other instances, well-known or trivial features such as user interface design are not described in detail so as not to obscure the present invention.

FIG. 1 illustrates a high level schematic block diagram of a host computer 100 connected with an external portable token 110 using communication link 120. The communication link 120 represents the media through which data is communicated between the portable token 110 and the host computer 100 via

the token interface 130 and the computer interface 140. The communication link includes but is not limited to any communication module or media such as radio frequency (RF) channels, or parallel connections. The external portable token 110 could be any peripheral device such as USB flash drive, a mobile phone or even another computer. The host computer 100 is any computing machine such as desktop computer, notebook computer or palmtop computer. The host computer contains a Virtual Disk Manager (VDM) 150.

To simplify the description, we will sometimes simply refer the external portable token as token.

FIG. 2 is a high level schematic block diagram showing some important components internal to the external portable token. The token contains Re-writable Nonvolatile Memory 200, an Embedded Controller (EC) 210 and the Computer Interface 140.

FIG. 3 shows some important data components inside the re-writable nonvolatile memory. Token Batch Key 310 is a random sequence of bits. The same Token Batch Key is stored in every token in a given batch. The nonvolatile memory also contains other data items such as Name 320, Password 330, Counter 340 and Token Key 350. The Name 320 is used to identify a particular batch of tokens. The Password 330 is used to control user access to the token. The Counter 340 is a small positive integer used to control the number of consecutive failed password retries. The counter value decreases by 1 for each incorrect password input and the token will be disabled when the counter value reaches zero. However, the Counter will be reset to the original value when a correct password input is detected. The Token Key 350 is a random secret key used for file encryption and decryption under a symmetric key encryption algorithm.

FIG. 4 shows the operation of VDM 150 during system initialization. To initialize a token, a user connects his/her token to the host computer in step 400. Upon detecting a user request to initialize the token in step 410, VDM asks the user to input a Name 320, a Password 330, a Counter 340 in step 420. The VDM then generates a random Token Key 350 in step 430, and stores Name, Password, Counter and Token Key into the token in step 440. Optionally, the Token Key 350 can be generated inside the token itself.

A user may have a batch of tokens which all contain the same Token Batch Key 310. Having more than one token allows the user to back up his/her critical data in multiple tokens. In case a token is lost, he/she is still able to access his/her confidential files stored in the host computer using other tokens. The token back up procedure is as follows. First, the user connects the token to be backed up to the host computer. Next, the Name 320, Password 330, Counter 340 and Token Key 350 are encrypted inside the token using Token Batch Key 310. The resulting ciphertext is then transfer to the host computer. The user disconnects the token to be backed up and connects a new token to the host computer. Next,

the host computer transfers the ciphertext to the new token. Finally, the ciphertext is decrypted inside the new token using his Token Batch Key and the resulting plaintext is stored in the re-writable nonvolatile memory of the new token.

FIG. 5 depicts the operation of the VDM during creation of a secure virtual disk. A user request to create a secure virtual disk is received in step 500. The user is promoted to plug in his/her token and key in his/her password in step 510. The correctness of the password is checked in step 520. If the password is incorrect, the operation is aborted in step 530. If the password is correct, the process continues to step 540 where the user is asked to specify the name, size and location of the secure virtual disk. Based on the user's specifications, a computer file is created and then mapped to a virtual disk in step 550.

The VDM performs access control to the secure virtual disk based on four events: token plug-in, token removal, user login and user logout. A user is allowed access to the secure virtual disk only if the token is plugged in and the login password is correct. During user access to the secure virtual disk, the VDM constantly detects if the token is present. If VDM detects that the token is removed from the host computer, the user is immediately denied access to the secure virtual disk.

Referring to FIG. 6 where at step 600 a user requests access to the secure virtual disk. Upon receiving the request, the VDM asks the user to plug in his/her token and key in his/her password at step 610. The validity of the password is checked in step 620. If the password is incorrect, the operation aborts at step 630. If the password is correct, the operation proceeds at step 640 where the secure virtual disk is mapped to a virtual disk for user access.

Files written to the secure virtual disk is encrypted using a symmetric key encryption algorithm. FIG. 7 shows the encrypted file structure which consists of a File Header 710 and a File Body 720. The File Header 710, encrypted using the Token Key 350, consists of File Attributes and a File Encryption Key. Encryption of the File Header is performed by the EC 210 inside the token.

The File Body 720 is encrypted using the File Encryption Key with a symmetric key encryption algorithm. This encryption is performed by the VDM 150 in the host computer.

The file encryption process is shown in FIG. 8. The VDM accepts a user request to write a file to the secure virtual disk in step 800. In step 810, the VDM creates a file header and sends it to the token for encryption, receives the encrypted file header from the token, and then writes the ciphertext to the secure virtual disk. At step 820, the VDM encrypts the file body and writes the ciphertext to the secure virtual disk.

The file decryption process is given in FIG. 9. The VDM accepts a user request to read a file from the secure virtual disk at step 900. At step 910, the VDM reads the encrypted file header, sends it to the token for decryption, and receives from the token the plaintext file header. At step 920, the VDM reads the encrypted file body and decrypts it using the file encryption key obtained from the plaintext file header.

While the present invention has been described in terms of specific embodiments, it is to be understood that the invention is not limited to these disclosed embodiments. This invention may be embodied in many different forms and should not be construed as limited to the embodiments set forth herein; rather, these embodiments are provided by way of illustration only and so that this disclosure will be thorough, complete and will fully convey the full scope of the invention to those skilled in the art. Indeed, many modifications and other embodiments of the invention will come to mind of those skilled in the art to which this invention pertains, and which are intended to be and are covered by both this disclosure, the drawings and the claims.

What is claimed is:

1. A system for protecting user data privacy, said system including at least one user with a batch of external portable tokens, and at least one host computer, a method for token initialization comprising of the steps of:
 - a. generating a batch token key and storing it in every token in the batch;
 - b. accepting a name, a password, a counter and a token key from a user and storing them in the re-writable nonvolatile memory of one token from the batch;
 - c. encrypting said name, password, counter and token key using batch token key and transferring the ciphertext into a new token in the batch;
 - d. said new token decrypting the ciphertext and storing name, password, counter and token key in its re-writable nonvolatile memory.
2. A system for protecting user data privacy as claimed in Claim 1, said system including at least one user with a batch of external portable tokens, and at least one host computer, a method for creating a secure virtual disk in the host computer comprising of the steps of:
 - a. accepting a user request for creating a secure virtual disk;
 - b. promoting said user to plug-in his/her token and key in his/her password;
 - c. promoting said user to specify name, size and location of the secure virtual disk;
 - d. creating the secure virtual disk according to the user specifications.
3. A system for protecting user data privacy as claimed in Claim 2, said system including at least one user with a batch of external portable tokens, and at least one host computer, a method for accessing a secure virtual disk in the host computer comprising of the steps of:
 - a. accepting user's request to access a secure virtual disk;
 - b. promoting said user to plug-in a token and input his/her password;
 - c. mapping the secure virtual disk into a virtual disk for user access;
 - d. constantly monitoring the presence of the token; once token removal is detected, denying the secure virtual disk from further access by said user.

4. A system for protecting user data privacy as claimed in Claim 3, said system including at least one user with a batch of external portable tokens, and at least one host computer, a method for writing a file into a secure virtual disk in the host computer comprising of the steps of:
 - a. accepting user request to write a file into a secure virtual disk;
 - b. creating a file header consisting of file attributes and a file encryption key;
 - c. sending file header to token for encryption, where the file header is encrypted using said token's token key;
 - d. receiving encrypted file header from said token and writing it to the secure virtual disk;
 - e. encrypting file body using file encryption key and writing it to the secure virtual disk.
5. A system for protecting user data privacy as claimed in Claim 4, said system including at least one user with a batch of external portable tokens, and at least one host computer, a method for reading a file from a secure virtual disk in the host computer comprising of the steps of:
 - a. accepting user request to read a file from a secure virtual disk;
 - b. reading encrypted file header and sending it to said token for decryption using the token key;
 - c. receiving decrypted file header and obtaining a file encryption key from the header;
 - d. reading encrypted file body from said secure virtual disk;
 - e. decrypting encrypted file body using said file encryption key.