ASYMMETRIC CRYPTOGRAPHY WITH ROLLING KEY SECURITY

Inventors: Ravinderpal Singh Sandhu, Oak Hill, VA (US); Brett Jason Schoppert, Leesburg, VA (US); Ravi Ganesan, Half Moon Bay, CA (US); Mihir Bellare, San Diego, CA (US); Colin Joseph deSa, Herndon, VA (US)

Assignee: VMware, Inc., Palo Alto, CA (US)

Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 1422 days.

Appl. No.: 11/332,204
Filed: Jan. 17, 2006

Prior Publication Data

Related U.S. Application Data
Provisional application No. 60/644,028, filed on Jan. 18, 2005.

Int. Cl.
G06F 21/00 (2006.01)
H04L 9/08 (2006.01)

U.S. Cl. ....... 713/189; 713/193; 713/194; 380/286; 380/277; 380/278; 726/26

Field of Classification Search 380/286
See application file for complete search history.

References Cited
U.S. PATENT DOCUMENTS

5,485,519 A 1/1996 Weiss
5,737,419 A * 4/1998 Ganesan ................. 713/169
5,807,578 A 2/1999 Brickell et al.
5,920,630 A * 7/1999 Wertheimer et al. ........ 380/286

OTHER PUBLICATIONS

Abstract
A system for securing information, includes a processor and storage device. The storage device stores information encrypted with one of a first private rolling key and a first public rolling key of an asymmetric rolling crypto-key, along with the one first rolling key. The processor has the logic to direct transmission, via a network, of knowledge of the stored first rolling key to authenticate a user, and of a request for the other of the first private rolling key and the first public rolling key. The processor receives the other first rolling key via the network, responsive to the directed transmission. The processor then decrypts the stored encrypted information with the received other first rolling key, and generates a second asymmetric rolling crypto-key having a second private rolling key and a second public rolling key. The processor encrypts the information with one of the second private rolling key and the second public rolling key. The processor also directs transmission of the other of the second private rolling key and the second public rolling key via the network. The storage device stores the information encrypted with the one second rolling key and the one second rolling key itself.

15 Claims, 11 Drawing Sheets
### U.S. PATENT DOCUMENTS

<table>
<thead>
<tr>
<th>Patent Number</th>
<th>Date</th>
<th>Inventor(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5,953,422 A</td>
<td>9/1999</td>
<td>Angelo et al.</td>
</tr>
<tr>
<td>6,026,163 A</td>
<td>2/2000</td>
<td>Miceli et al.</td>
</tr>
<tr>
<td>6,109,802 B1</td>
<td>1/2001</td>
<td>Lerner et al.</td>
</tr>
<tr>
<td>6,212,635 B1</td>
<td>4/2001</td>
<td>Reardon et al.</td>
</tr>
<tr>
<td>6,608,901 B2</td>
<td>8/2003</td>
<td>Scheidt et al.</td>
</tr>
<tr>
<td>6,662,299 B1</td>
<td>12/2003</td>
<td>Price, III</td>
</tr>
<tr>
<td>6,697,488 B1</td>
<td>2/2004</td>
<td>Cramer et al.</td>
</tr>
<tr>
<td>6,845,160 B1</td>
<td>1/2005</td>
<td>Aoki et al.</td>
</tr>
<tr>
<td>7,095,851 B1</td>
<td>8/2006</td>
<td>Scheidt</td>
</tr>
<tr>
<td>2006/0182277 A1</td>
<td>8/2006</td>
<td>Sandhu</td>
</tr>
<tr>
<td>2006/0182283 A1</td>
<td>8/2006</td>
<td>Sandhu</td>
</tr>
</tbody>
</table>

### OTHER PUBLICATIONS


* cited by examiner
FIG. 1
FIG. 5
Network Device 1
[e.g. User Network Device]

Generate Factor $F_K$
For Split Asymmetric
CRYPTO-KEY
[e.g. Generate $D_{USB}, E_{USB}, N_{USB}$]

Generate Rolling
Asymmetric CRYPTO-KEY
For Securing $F_K$
[e.g. Generate $D_R, E_R, N_R$]

Secure Factor $F_K$
with Rolling Key
[e.g. Encrypt $(D_{USB}, D_R)$]

Encrypt Rolling Key
[e.g. Encrypt $(D_R, K_{DR})$]
(OPTIONAL)

Store Secured Factor $F_K$ and Optionally
Encrypted Rolling Key
[e.g. Store $D_{USB} + D_R$]

Transmit Other Rolling
Key and Factor Related
Information
[e.g. Transmit $E_R, N_R + E_{USB}, N_{USB}$]

To 630

Network Device 2
[e.g. Sponsor Station]

Encrypt Other Rolling
Key and Factor Related
Information
[e.g. Encrypt $(E_R, K_{ER}) +$
Encrypt $(E_{USB}, K_{EUSB})$]
(OPTIONAL)

Store Optionally
Encrypted Other
Rolling Key and Factor
Related Information
[e.g. Store $E_R, N_R + E_{USB}, N_{USB}$]

Encrypt Other
Split Key Portion
[e.g. Encrypt $(D_{U2}, K_{DU2})$
$+ Encrypt (E_U, K_{EU})$]
(OPTIONAL)

Store Optionally
Encrypted Other
Split Key Portion
[e.g. Store $D_{U2}, E_U, N_U$]
Network Device 1
[e.g. User Network Device]

630

Destroy Other Rolling Key and Factor Related Information
[e.g. Destroy $E_R$, $N_R$ + $E_{USB}$, $N_{USB}$]

635

Generate User's Asymmetric CRPTO-KEY
[e.g. Generate $D_U$, $E_U$, $N_U$]

640

Split a Key of User's Asymmetric CRYPTO-KEY
[e.g. Split $D_U$ into $D_{U1} + D_{U2}$]

645

Compute Split Key Portion Based on Factor $F_K$ and another Factor
[e.g. Compute $D_{U1}$ Based on $D_{USB} + F_P$]

650

Compute Other Split Key Portion
[e.g. Compute $D_{U2}$ Based on Computed $D_{U1}$]

655

Destroy Split Key Portion and Split Key
[e.g. Destroy $D_U + D_{U1}$]

To 660

FIG. 6B
Network Device 1
[e.g. User Network Device]

 Transmit Other Split Key Portion
[e.g. Transmit $D_{U2}, E_U, N_U$]

Destroy User's Public Key
[e.g. Destroy $E_U, N_U$]
(Optional)

To 662

FIG. 6C
FIG. 7B
Network Device 1
[e.g. User Network Device]

Sign Permission Request with Computed Split Key Portion and Decrypted Factor $F_K$
[e.g. Encrypt (PR, $D_{USB}$) $D_U$]

Transmit Signed Permission Request
[e.g. Transmit CSPR]

Network Transmission

Network Device 2
[e.g. Sponsor Station]

Recover Permission Request Based on Transmitted Split Key Portion
[e.g. Decrypt (CSPR, $D_{USB}$) $E_U$]

810

Permission Request Recovered?

815

No

Suspend User ID

Yes

Recover Permission Request Based on Transmitted Decrypted Factor $F_K$
[e.g. Decrypt (SPR, $D_{USB}$), $E_{USB}$]

820

Permission Request Recovered?

825

No

Suspend User ID

Yes

Login Completed

FIG. 8
ASYMMETRIC CRYPTOGRAPHY WITH ROLLING KEY SECURITY

RELATED APPLICATIONS

This application claims priority based upon Provisional U.S. Application Ser. No. 60/644,028, filed Jan. 18, 2005, and entitled “THE TRICIPHER ARMORED CREDENTIAL SYSTEM”, the contents of which are incorporated herein in their entirety by reference.


TECHNICAL FIELD

This invention relates to cryptosystems. More particularly, the present invention relates to asymmetric cryptography with rolling key security.

BACKGROUND ART

Today, computing devices are almost always interconnected via networks. These networks can be large closed networks, as within a corporation, or truly public networks, as with the Internet. A network itself might have hundreds, thousands or even millions of potential users. Consequently it is often required to restrict access to any given networked computer or service, or a part of a networked computer or service, to a subset of the users on the public or closed network. For instance, a brokerage might have a public website accessible to all, but would like to only give Ms. Alice Smith access to Ms. Alice Smith’s brokerage account.

Access control is an old problem, tracing its roots to the earliest days of computers. Passwords were among the first techniques used, and to this day remain the most widely used, for protecting resources on a computer or service.

Single-Factor Authentication

In its simplest form, known as single factor authentication, every user has a unique password and the computer has knowledge of the user password. When attempting to log on, Alice would enter her userid, say alice, and password, say apple23, the computer would compare the pair, i.e. alice, apple23, with the pair it had stored for Alice, and if there is a match would establish a session and give Alice access.

This simple scheme suffers from two problems. First, the table containing the passwords is stored on the computer, and thus represents a single point of compromise. If Eve could somehow steal this table, she would be able to access every user’s account. A second problem with this approach is that when Alice enters her password it travels from her terminal to the computer in the clear, and Eve could potentially eavesdrop. Such eavesdropping is known as a Man-In-The-Middle attack. For instance the terminal could be Alice’s PC at home, and the computer could be a server on the Internet, in which case her password travels in the clear on the Internet. It will be recognized by those with ordinary skill in the art that a Man-In-The-Middle attack can go beyond eavesdropping, to modifying the contents of the communication.

Various solutions have been proposed and implemented to address these two issues. For instance, to address the first problem of storing the password on the computer, the computer could instead store a one way function of the password, e.g. F(apple23) → XD45DTY, and the pair [alice, XD45DTY]. In this example as F(apple23) is a one way function, computing XD45DTY from apple23 is easy, but as it is a “one way function”, the reverse is believed to be computationally difficult or close to impossible. So when Alice logs on and sends the computer [alice, apple23], the computer can compute F(apple23) and compare the result with XD45DTY. The UNIX operating system was among the first to implement such a system in the 1970’s. However, this approach, while addressing the problems due to the storage of the password on the computer, does not address the problem of the password traveling in the clear.

Multifactor Authentication

Multiple factor authentication also exists as a potential solution to the problems inherent with single factor authentication. In multiple factor authentication, at least knowledge of, if not actual possession of, two or more factors must be shown for authentication to be complete. It should be understood that in multiple factor authentication, each factor remains separate. That is, the factors are not combined. Further, the factors are not even concatenated. Several multiple factor authentication techniques exist, including one time password token techniques, encrypted storage techniques, smart card techniques, and split key techniques.

In one time password token techniques, two passwords are utilized, one being a permanent password associated with the user, and the other being a temporary, one-time use, password generated by a password generator. The permanent password
may be optional. The temporary password has a finite usable life, such as sixty seconds. At the end of the usable life, another temporary password is generated. An authentication server knows each usable password as well as its usable life, based upon algorithms well known to those of ordinary skill in the art. A user transmits both the permanent password (first factor) and a temporary password (second factor) to the authentication server which then verifies both passwords. The passwords are transmitted in the clear, thus token techniques are subject to man-in-the-middle attacks.

Storage of Crypto-Keys

Using encrypted storage techniques, a cryptographic key is stored on either removable media or a hard drive. The cryptographic key is encrypted with a user’s password. After decryption with the user’s password, the key is then stored, at least temporarily, in memory of the user’s computer system where it is used to encrypt or decrypt information. As will be recognized by those of ordinary skill in the art, this particular approach is undesirable due to it being susceptible to a dictionary attack, to be discussed in detail further below.

In smart card techniques, a private portion of an asymmetric cryptographic key is stored on a smart card, which is portable. A specialized reader attached to a computer system is used to access the smart card. More particularly, the user enters a PIN (the first factor) to ‘unlock’ the smart card. Once unlocked, the smart card encrypts or decrypts information using the key stored thereon. It should be stressed that in smart card techniques the key never leaves the smart card, unlike in the encrypted storage techniques discussed above. Rather, electronics within the smart card itself perform the encrypting and/or decrypting. Smart card techniques are associated with certain problems. These problems include the fact that the technique is costly to implement. Due to hardware costs. Further, a lack of readers makes use of a user’s smart card difficult, and smart cards themselves are subject to loss.

Symmetric and Asymmetric Cryptography

Before discussing in detail the more sophisticated conventional techniques for authentication, which are based upon split key technology, let us briefly describe symmetric and asymmetric key cryptography.

In symmetric key cryptography, the two parties who want to communicate in private share a common secret key, say $K$. The sender encrypts a message with $K$, to generate a cipher, i.e. $C = \text{Encrypt}(K, M)$. The receiver decrypts the cipher to retrieve the message, i.e. $M = \text{Decrypt}(C, K)$. An attacker who does not know $K$, and sees $C$, cannot successfully decrypt the message $M$, if the underlying algorithms are strong. Examples of such systems are DES3 and RC4. Encryption and decryption with symmetric keys provide a confidentiality, or privacy service.

Symmetric keys can also be used to provide integrity and authentication of messages in a network. Integrity and authentication means that the receiver knows who sent a message and that the message has not been modified so it is received as it was sent. Integrity and authentication is achieved by attaching a Message Authentication Code (MAC) to a message $M$. E.g., the sender computes $S = \text{MAC}(M, K)$ and attaches $S$ to the message $M$. When the message $M$ reaches the destination, the receiver also computes $S' = \text{MAC}(M, K)$ and compares $S'$ with the transmitted value $S$. If $S' = S$ the verification is successful, otherwise verification fails and the message should be rejected. Early MAC’s were based on symmetric encryption algorithms such as DES, whereas more recently MACs are constructed from message digest functions, or “hash” functions, such as MD5 and SHA-1. The current Internet standard for this purpose is known as hash-based MAC (HMAC).

By combining confidentiality with integrity and authentication, it is possible to achieve both services with symmetric key cryptography. It is generally accepted that different keys should be used for these two services and different keys should be used in different directions between the same two entities for the same service. Thus if Alice encrypts messages to Bob with a shared key $K$, Bob should use a different shared key $K'$ to encrypt messages from Bob to Alice. Likewise Alice should use yet another key $K''$ for MACs from Alice to Bob and Bob should use $K''$ for MACs from Bob to Alice. Since this is well understood by those skilled in the art, we will follow the usual custom of talking about a single shared symmetric key between Alice and Bob, with the understanding that strong security requires the use of four different keys.

Symmetric key systems have always suffered from a major problem—namely how to perform key distribution. How do Bob and Alice agree on $K$?

Asymmetric key cryptography was developed to solve this problem. Here every user is associated with a private/public key pair, commonly referred to as $D$ and $E$, which are related by special mathematical properties. These properties result in the following functionality: a message encrypted with one of the two keys can then only be decrypted with the other.

One of these keys for each user is made public and the other is kept private. Let us denote the former by $E$, and the latter by $D$. So Alice knows $D_{alice}$ and everyone knows $E_{alice}$. To send Alice the symmetric key $K$, Bob simply sends ciphertext $C = \text{Encrypt}(K, E_{alice})$. Alice, and only Alice (since no one else knows $D_{alice}$), can decrypt the ciphertext $C$ to recover the message, i.e. $\text{Decrypt}(C, D_{alice}) = K$. Now both Alice and Bob know $K$ and can use it for encrypting subsequent messages using a symmetric key system. Why not simply encrypt the message itself with the asymmetric system? This is simply because in practice all known asymmetric systems are fairly inefficient, and while they are perfectly useful for encrypting short strings such as $K$, they are inefficient for large messages.

The above illustrates how asymmetric cryptography can solve the key distribution problem. Asymmetric cryptography can also be used to solve another important problem, that of digital signatures. To sign a message $M$, Alice encrypts it with her own private key to create $S = \text{Encrypt}(M, D_{alice})$. She can then send $(M, S)$ to the recipient who can then decrypt $S$ with Alice’s public key to generate $M'$, i.e. $M' = \text{Decrypt}(S, E_{alice})$. If $M' = M$ then the recipient has a valid signature as only someone who has $D_{alice}$, by definition only Alice, can generate $S$, which can be decrypted with $E_{alice}$ to produce $M$. To convey the meaning of these cryptographic operations more clearly they are often written as $S = \text{Sign}(M, D_{alice})$ and $M' = \text{Verify}(S, E_{alice})$. It is worth noting that asymmetric key digital signatures provide non-repudiation in addition to the integrity and authentication achieved by symmetric key MACs. With MACs the verifier can compute the MAC for any message $M$ of his choice, since the computation is based on a shared secret key. With digital signatures this is not possible since only the sender has knowledge of the sender’s private key required to compute the signature. The verifier can only verify the signature, but not generate it. It will be recognized by those skilled in this art that there are numerous variations and elaborations of these basic cryptographic operations of symmetric key encryption, symmetric key MAC, asymmetric key encryption and asymmetric key signatures.

The RSA cryptosystem is one system that implements asymmetric cryptography as described above. In particular the RSA cryptosystem allows the same private-public key pair to be used for encryption and for digital signatures. It should be noted that there are other asymmetric cryptosystems that implement encryption only e.g., ElGamal, or digital
signature only, e.g., DSA. Technically the public key in RSA
is a pair of numbers E, N and the private key is the pair of
numbers D, N. When N is not relevant to the discussion, it is
commonplace to refer to the public key as E and the private
key as D.

Finally, the above description does not answer the impor-
tant question of how Bob gets Alice’s public key E_{Alice}. The
process for getting and storing the binding [Alice, E_{Alice}]
which binds E_{Alice} to Alice is tricky. The most practical
method appears to be to have the binding signed by a common
trusted authority. So such a “certificate authority” (CA) can
create CERT_{Alice} := Sign(Alice, E_{Alice}, DCA). Now CERT_{Alice}
can be verified by anyone who knows the CA’s public key
ECA. So in essence, instead of everyone having to know every-
one else’s public key, everyone only need know a single
public key, that of the CA. More elaborate schemes with
multiple Certificate Authorities, sometimes having a hierar-
chy of responsibility, have also been proposed.

Asymmetric key cryptosystems have been around for a
long time, but have found limited use. The primary reasons
are twofold: (a) the private key D in most systems is long,
which means that users cannot remember them, and they have
to be stored on every computer they use, or carried around
on smart cards or other media; and (b) the infrastructure
for ensuring a certificate is valid, which is critical, is
cumbersome to build, operate, and use. The first technique
proposed to validate certificates was to send every recipient
a list of all certificates that had been revoked. This clearly
does not scale well to an environment with millions of users.
A later technique proposed to validate certificates was to require
that one inquire about the validity of a certificate on-line,
which has its own associated problems.

Split Private Key Asymmetric Cryptography

A system based on split private key asymmetric crypto-
graphy has been developed to solve these two issues, i.e. long
private keys and certificate validity, among others. In this
system the private key for Alice, i.e. D_{Alice}, is further split into
two parts, a part D_{Alice} which Alice knows, and a part D_{SA}
which is stored at a security server, where D_{SA} \cdot D_{SA} = D_{Alice} \mod F(N).
To sign a message, Alice could perform a partial encryption
to generate a partial signature, i.e. PS = Sign(M, D_{SA}),
Alice then sends the server PS which completes the signature
by performing S = \text{Sign}(PS, D_{SA}). This completed signature
S is indistinguishable from one generated by the original
private key, i.e. D_{Alice}, so the rest of the process works as
previously described. However, D_{SA} can be made short, which
allows the user to remember it as a password, so this system
is user friendly. Further, if the server is informed that a par-
ticular ID has been suspended or revoked, then it will cease to
perform its part of the operation for that user, and conse-
quently no further signatures can ever be performed. This
provides for instant revocation in a simple, highly effective
fashion. It will be recognized by those skilled in the art that a
split private key can be used in a similar manner for decryp-
tion purposes, and that the partial signatures (or encryptions)
may be performed in the reverse sequence, that is first by the
security server and subsequently by the user’s computer,
or may even be performed concurrently in both places and then
combined.

Password Based Cryptography

Let us return now to password based systems. Challenge-
response systems solve the issue of having to send passwords
in the clear across a network. If the computer and Alice share
a secret password, P, then the computer can send her a new
random challenge, R, at the time of login. Alice computes
C = Encrypt(R, P) and sends back C. The computer decrypts
Decrypt(C, P) = C. If R = C, then the computer can trust that it
is Alice at the other end. Note however that the computer had
to store P.

A more elegant solution can be created using asymmetric
cryptography. Now Alice has a private key D_{Alice}, or in a split
private key system she has D_{SA}. The computer challenges her
to sign a new random challenge R. She signs the challenge, or
in the split private key system she interacts with the security
server to create the signature, and sends it back to the com-
puter which uses her public key, retrieved from a certificate, to
verify the signature. Observe that the computer does not have
to know her private key, and that an eavesdropper observing
the signature on R gains no knowledge of her private key.

The SSL system, which is widely used on the Internet, in
effect implements a more elaborate method of exactly this
protocol. SSL has two components, ‘server side SSL’ in
which a server proves its identity by correctly decrypting a
particular message during connection set-up. As browsers
such as Netscape and Microsoft Internet Explorer come
loaded with the public keys of various CAs, the browser can
verify the certificate of the server and use the public key
therein for encryption. This authenticates the server to the
client, and also allows for the set-up of a session key K, which
is used to encrypt and MAC all further communications.

Server side SSL is widely used, as the complexity of man-
aging certificates rests with system administrators of web sites
who have the technical knowledge to perform this function.
The converse function in SSL, client side SSL, which lets a
client authenticate itself to a server by means of a digital
signature is rarely used, because although the technical
mechanism is much the same, it now requires users to manage
certificates and long private keys which has proven to be
difficult, unless they use the split private key system. So in
practice, most Internet web sites use server side SSL to
authenticate themselves to the client, and to obtain a secure
channel, and from then on use UserId, Password pairs to
authenticate the client.

So far from disappearing, the use of passwords has
increased dramatically. Passwords themselves are often
dubbed as inherently “weak”. This is inaccurate because, if
they are used carefully, passwords can actually achieve
“strong” security. As discussed above, passwords should not be
sent over networks, and if possible should not be stored on
the receiving computer. Instead, in a “strong” system, the user
can be asked to prove knowledge of the password without
actually revealing the password. Perhaps most critical is that
passwords should not be vulnerable to dictionary attacks.

Dictionary Attacks

Dictionary attacks can be classified into three types. In all
three types the starting point is a ‘dictionary’ of likely
passwords. Unless the system incorporates checks to prevent it,
users tend to pick poor passwords, and compilations of lists of
commonly used poor passwords are widely available.

On line dictionary attack: Here the attacker types in a guess
at the password from the dictionary. If the attacker is granted
access to the computer they know the guess was correct.
These attacks are normally prevented by locking the user
account if there are an excessive number of wrong tries to gain
access. Note that this very commonly used defense prevented
one problem, but just created another one. An attacker can
systematically go through and lock out the accounts of hun-
dreds or thousands of users. Although the attacker did not
gain access, now legitimate users cannot access their own
accounts either, creating a denial of service problem.

Encrypt dictionary attacks: If somewhere in the operation of
the system a ciphertext C = Encrypt(M, P) was created, and
the attacker has access to both C and M, then the attacker can
compute off-line \text{C1} = \text{Encrypt(M, G1)}, \text{C2} = \text{Encrypt(M, G2)}, \ldots \text{where G1, G2, \ldots etc. are the guesses at the password P from the dictionary}. \text{The attacker stops when he finds a C = C, and knows that Gn = P. Observe that the UNIX file system, which uses a one way function F( ) instead of an encryption function E( ), is vulnerable to this attack.}

\text{Decrypt dictionary attacks: Here the attacker, does not know M, and only sees the ciphertext C (where C = Encrypt(M, P)). The system is only vulnerable to this attack if it is true that M has some predictable structure. So the attacker tries M1 = Decrypt(C, G1), M2 = Decrypt(C, G2), \ldots \text{and stops when the M1 has the structure he is looking for. For instance M1 could be known to be a timestamp, English text, or a number with special properties such as a prime, or a composite number with no small factors. Those with ordinary skill in the art will recognize there are numerous variations of the encrypt and decrypt dictionary attacks.}

\text{In split private key systems, the user portion of the private key, referred to as D_{priv} above, may come from the user’s password only. Thus, a compromise of the password, i.e., another person learning a user’s password, results in a compromise of the split private key system. Also, there still remains the possibility of a dictionary attack on the server portion of the private key, referred to as D_{priv}, because the user portion of the private key comes from the user’s password only. Thus knowledge of D_{priv} enables a dictionary attack on D_{server}. As discussed above, many of the existing multiple factor systems that address these problems rely upon expensive hardware. Because of this and other reasons, such systems have failed to gain widespread support. However, as will be discussed further below, recently a multifactor cryptographic system was developed which overcomes these problems.}

\text{Multifactor Split Private Key Asymmetric Cryptography}

\text{In particular, as disclosed in U.S. application Ser. No. 11/055,987, filed Feb. 14, 2005, and entitled “Architecture For Asymmetric Crypto-key Storage”, to overcome these problems Tricipher, Inc, the assignee of all rights in the present application, has developed an asymmetric cryptosystem in which users are associated with an asymmetric crypto-key having a public key and a private key split into multiple private key portions. As in the conventional split key asymmetric cryptosystems discussed above, each of the private key portions can be applied to an original message separately or in sequence and the partial results combined to form a transformed, i.e. encrypted, message, and the public key can be applied to the transformed message to verify authenticity of the message preferably by recovering the original message, which authenticates the user. Conversely a message transformed, i.e. encrypted, with the public key can be decrypted by applying each of the private key portions to the transformed message separately or in sequence and the partial results combined to decrypt the original message.}

\text{However, unlike the conventional split key asymmetric cryptosystem discussed above, the Tricipher system generates at least one of the multiple private key portions of the asymmetric crypto-key using multiple pieces of information, known as factors. For purposes of the following discussion, we will assume that a first private key portion of the asymmetric crypto-key is generated using multiple factors. The multiple factors could be two factors, three factors, or any number of multiple factors, as may be desired under the circumstances. In any event, each of the multiple factors is under the control of a single entity. That is, the single entity has possession of, or free access to, each of the multiple factors. For purposes of the following discussion, we will assume that the entity associated with the first private key portion is a user. Thus, the first private key portion could be D_{server}.}

\text{A factor could be as simple as a readily available number string, such as a serial number of a user’s computer, or could be a sophisticated algorithm, such as a cryptographic key. Preferably, one of the factors corresponds to the user’s password. If each of the multiple factors is a number string, generation of the first private key portion could be accomplished by simply concatenating the multiple factors. However advantageously, the first private key portion is generated by cryptographically combining the multiple factors, and each of the multiple factors is, or is used to produce, a cryptographic key. Thus, cryptographic keys are beneficially used to produce a cryptographic key.}

\text{The first private key portion is not stored in a persistent state. That is, the first private key portion must be generated whenever its use is desired. The first private key portion may be immediately destroyed after its initial use, or may be stored temporarily after its generation, making it available for use multiple times before it is destroyed, for example for use during a predetermined time period or for use a predetermined number of times.}

\text{Another of the multiple private key portions of the asymmetric crypto-key, which will be referred to as the second private key portion for purposes of this discussion, is under control of another entity, in this case an entity other than the applicable user, e.g. a secure server or another user. Thus, the second private key portion could be D_{server}. This second private key portion may be stored in a persistent state. In this example, it is assumed that the first and second private key portions of the asymmetric crypto-key are combinable to form a complete private key D_{server}. This private key is usable to transform, e.g. encrypt or decrypt, messages as may be desired under the circumstances.}

\text{Thus, in the multifactor split private key asymmetric cryptosystem developed by Tricipher, Inc, one of multiple private key portions of an asymmetric crypto-key, e.g. the user’s private key portion, is generated using multiple factors, with each of these factors being under the control of the applicable entity, e.g. that user. This results in a private key that provides greater security than the private keys in other split private key asymmetric cryptosystems in which each entity’s, e.g. each user’s, private key portion is generated using only a single factor.}

\text{Additionally, in the multifactor split private key asymmetric cryptosystem developed by Tricipher, Inc, preferably some, but not all, of the multiple factors used to generate the user’s private key portion may be stored, with each stored in a in different location.}

\text{For example, if one of the factors corresponds to the user’s password, preferably neither the user’s password nor the corresponding factor is stored, except temporarily on random access memory (RAM) for only the time necessary, in the case of the password, to allow for generation of the corresponding factor after the user enters the password and, in the case of the corresponding factor, to allow for generation of the first private key portion after the corresponding factor has been generated. On the other hand, the other of the multiple factors may be stored. Thus, if there are two other factors, these later two factors are preferably stored separately at different locations. This adds a level of security, in that a thief would have to infiltrate two locations to steal both of these factors.}

\text{In this regard, one of these later factors may be stored on either a user’s computing device or removable media configured to communicate with the user’s computing device. As}
will be recognized by those skilled in the art, the user’s computing device could be a personal computer (PC), personal computing device, mobile phone or some other type of computing device, and the removable media could be a USB flash drive, smart card, floppy disk, compact disk (CD) or some other type of portable data storage device. A factor stored on a user’s computing device is sometimes referred to as D\textsubscript{other}, and a factor stored on removable media is sometime referred to as D\textsubscript{USB}.

The first private key portion is temporarily stored after being generated. During this temporary storage, it is usable to prove the user’s identity multiple times without the user providing any authentication information. This temporary storage could be limited to a predefined time period, or a predefined number of authentications.

The public key, commonly designated as E, is preferably stored under control of an entity other than the entity having access to the multiple factors, e.g., other than the applicable user in the example above. Thus, the public key is available to at least one other entity.

In the multifactor split private key asymmetric cryptosystem developed by Tricipher Inc., another of the private key portions of the split private key may also be generated based on the same or other multiple factors, which could be partially or completely under the control of the same or another entity, e.g., could be under the control of the applicable user or another entity in the example above. In any event, this other private key portion could be generated based on multiple factors in same manner as described above for the first private key portion.

As described in the above referenced application, in the multifactor split private key asymmetric cryptosystem developed by Tricipher Inc., the factor D\textsubscript{other} is preferably the private key of a private-public asymmetric key pair, including D\textsubscript{other} and E\textsubscript{other} and having a modulus N\textsubscript{other}. The factor D\textsubscript{USB} is preferably the private key of a private-public asymmetric key pair, including D\textsubscript{USB}, and E\textsubscript{USB} and having a modulus N\textsubscript{USB}. For example, D\textsubscript{other} and E\textsubscript{other} and/or D\textsubscript{USB} and E\textsubscript{USB} may form a conventional RSA private-public asymmetric key pair.

The non-private parts of the generated keys, i.e., E\textsubscript{other}, N\textsubscript{other}, E\textsubscript{USB}, and N\textsubscript{USB} are stored, for example, at a trusted server. In the most common implementation, D\textsubscript{other} is stored securely on the hard disk of a user’s PC using the protection capabilities provided by the PC’s operating system, preferably as a non-exportable private key in a Windows Operating System key-store. Of course, as desired, D\textsubscript{other} could be stored in a Windows Operating System registry. Alternatively, D\textsubscript{other} can be, as desired, stored on the trusted processing module (TPM). No matter where or how on the user’s computing device D\textsubscript{other} is stored, in the most basic configuration, D\textsubscript{other} can only be used from the user’s computing device upon which it is stored. That is, D\textsubscript{other} is a non-exportable private key stored on the user device upon which it will be used. However, the multifactor split private key asymmetric cryptosystem developed by Tricipher, Inc. also facilitates porting D\textsubscript{other} to other devices for use thereon.

D\textsubscript{USB}, which is stored on removable media, also needs to be protected, since storing any kind of key in the clear should be avoided if possible. In the case of D\textsubscript{USB} this is particularly important because if D\textsubscript{USB} is stored on the removable media in clear and should the user misplace or otherwise lose the removable media an attacker could easily access, extract and/or copy D\textsubscript{USB} from the removable media, and potentially use D\textsubscript{USB} to impersonate the user. Thus, in the multifactor split key asymmetric cryptosystem developed by Tricipher Inc., D\textsubscript{USB} is beneficially stored on the removable media in an encrypted state.

Preferably, encryption of D\textsubscript{USB} is not performed with the user’s password, because this would still leave D\textsubscript{USB} vulnerable to dictionary attacks and guessing attacks. If the factors include both D\textsubscript{other} and D\textsubscript{USB}, D\textsubscript{USB} may, as disclosed in the above referenced application, be encrypted using D\textsubscript{other}, which is stored, for example, on the user’s PC.

However, if D\textsubscript{USB} is encrypted using D\textsubscript{other}, an attacker with access to the removable media, as well as the user’s PC or other computing device storing D\textsubscript{other} and the user’s password could obtain D\textsubscript{other} from the user’s computing device, decrypt D\textsubscript{USB} by applying D\textsubscript{other}, and then use the user’s password, D\textsubscript{USB} and D\textsubscript{other} to generate the first private key portion, e.g., D\textsubscript{other}. Having the first private key portion, the attacker would now be in a position to impersonate the user. Furthermore, in cases where the factors include D\textsubscript{USB} but not D\textsubscript{other} there is no tethering key to encrypt D\textsubscript{USB}.

Accordingly, a need exists for better techniques for securing stored factors used to generate multifactor split private keys in an asymmetric cryptosystem.

OBJECTIVES OF THE INVENTION

It is an object of the present invention to overcome the deficiencies of existing cryptosystems. Additional objects, advantages, novel features of the present invention will become apparent to those skilled in the art from this disclosure, including the following detailed description, as well as by practice of the invention. While the invention is described below with reference to a preferred embodiment(s), it should be understood that the invention is not limited thereto. Those of ordinary skill in the art having access to the teachings herein will recognize additional implementations, modifications, and embodiments, as well as other fields of use, which are within the scope of the invention as disclosed and claimed herein and with respect to which the invention could be of significant utility.

SUMMARY DISCLOSURE OF THE INVENTION

In accordance with the present invention a technique is provided for securing information on a communications network. The information is secured by storing the information encrypted with one of a first private rolling key and a first public rolling key of a first asymmetric rolling crypto-key, e.g. encrypted with the first private rolling key, and the one first rolling key itself, preferably in association with each other. The asymmetric rolling crypto-key could be an RSA or other type asymmetric crypto-key. The stored one first rolling key, e.g., the first private rolling key, is transmitted, via a network, such as the Internet, by a first entity to a second entity, to authenticate the first entity to the second entity. A request for the other of the first private rolling key and the first public rolling key, e.g., the first public rolling key, is also transmitted either together with or separate from the transmitted one first rolling key, e.g., the first private rolling key. The requested other first rolling key, e.g., the first public rolling key, is received by the first entity from the second entity via the network responsive to the transmission of the one first rolling key and the request. Typically, this will only occur if the first entity is authenticated based on the transmitted one first rolling key or otherwise demonstrates knowledge of this key, such as by signing a time stamp or challenge with the one first rolling key and transmitting the signed time stamp or challenge to the second entity. The stored informa-
tion encrypted with the one first rolling key, e.g. the first private rolling key, is decrypted by the first entity applying the received other first rolling key, e.g. the first public rolling key. After the information encrypted with the one first rolling key, e.g. the first private rolling key, is decrypted, a second asymmetric rolling crypto-key having a second private rolling key and a second public rolling key is generated by the first entity, and the information is encrypted by the first entity with one of the second private rolling key and the second public rolling key, e.g. encrypted with the second private rolling key. The second asymmetric rolling key could also be an RSA or other type asymmetric crypto-key. The information encrypted with the one second rolling key, e.g. with the second private rolling key, and the one second rolling key itself are stored so as to be accessible to the first network entity. Preferably, the other of the second private rolling key and the second public rolling key, e.g. the second public rolling key, is transmitted from the first entity to the second entity via the network, and the first entity does not retain a copy of this rolling key.

The first entity may generate the first asymmetric rolling key, and encrypt the information with the one first rolling key, e.g. with the first private rolling key. In such case, the stored information encrypted with the one first rolling key is the information generated and encrypted by the first network entity. However, it should be understood that another entity, e.g. the second entity referred to above, could alternatively, or in conjunction with first entity, generate each asymmetric rolling key, and encrypt the information with the one rolling key.

If the information is a factor for generating a private key of a private-public key pair of an asymmetric crypto-key associated with the first entity, the private key of the first entity’s asymmetric crypto-key could be generated by the first entity based on the decrypted factor. Furthermore, if the private key of the first entity’s asymmetric crypto-key is a split private key having a first private key portion and a second private key portion, the factor may be one of multiple factors for generating the first private key portion of the split private key. In such a case, the first entity may generate the first private key portion of the first entity’s asymmetric crypto-key based on the decrypted factor and another of the multiple factors corresponding to a password associated with the first entity.

Proof of knowledge of the generated first private key portion, such as a time stamp or challenge signed with the first private key portion, can then be transmitted from the first entity to the second entity via the network, to further authenticate the first entity to the second entity.

It should be understood that proof of knowledge of the stored one second rolling key, e.g. the second private rolling key, such as a time stamp or challenge signed with the one second rolling key, can be transmitted via the network from the first entity to the second entity to authenticate the first entity to the second entity. A request for the other of the second private rolling key and the second public rolling key, e.g. the second public rolling key, can also be transmitted via the network from the first entity to the second entity either with, or separately from, the transmitted proof of knowledge of the one second rolling key. In any event, the requested other second rolling key, e.g. the second public rolling key, is received via the network by the first entity from the second entity based on the transmitted proof of knowledge of the one second rolling key and the transmitted request. The stored information encrypted with the one second rolling key, e.g. encrypted with the second private rolling key, is then decrypted, by the first entity through the application of the received other second rolling key e.g. the second public rolling key. After the stored information encrypted with the one second rolling key, e.g. encrypted with the second private rolling key, is decrypted, the first entity can generate a third asymmetric rolling crypto-key having a third private rolling key and a third public rolling key, encrypt the information with one of the third private rolling key and the third public rolling key, e.g. with the third private rolling key, and store this encrypted information, so as to be accessible by the first entity.

A system implementation of the invention includes a storage device and processor. The storage device, e.g. a portable storage device, e.g. a removable medium such as a USB memory device, is configured to interconnect with the processor and store information encrypted with one of a first private rolling key and a first public rolling key of an asymmetric rolling crypto-key, as well as the first private rolling key itself, e.g. the first private rolling key. The asymmetric rolling crypto-key could be an RSA asymmetric crypto-key, or another type asymmetric crypto-key.

The processor is configured with logic to direct transmission, via a communications network such as the Internet, of proof of knowledge of the stored one first rolling key, e.g. the first private rolling key, to authenticate a user, and a request for the other of the first private rolling key and the first public rolling key. Most often, this transmission will be made in response to a user input indicating a desire to login, however this is not mandatory. It should be understood that the transmission of proof of knowledge of the stored one first rolling key and the request for the other first rolling key could be directed at the same time or different times, and the proof of knowledge and request could be transmitted separately or together. In any event, the processor receives the other first rolling key, e.g. the first public rolling key, via the network in response to the directed transmission(s). Typically, the processor will only receive the other first rolling key if the user can be authenticated based on the transmitted proof of knowledge of the first private rolling key, e.g. based on proof of knowledge of the first private rolling key if that is the key accessible to the processor. The processor can now decrypt the stored encrypted information with the received other first rolling key, e.g. with the first public rolling key.

After decrypting the information, the processor generates a second asymmetric rolling crypto-key having a second private rolling key and a second public rolling key, and encrypts the information with one of the second private rolling key and the second public rolling key, e.g. with the second private rolling key. Here again, the second asymmetric rolling crypto-key could be an RSA asymmetric crypto-key, or another type asymmetric crypto-key. The processor also directs transmission of the other of the second private rolling key and the second public rolling key via a network, e.g. the second public rolling key, preferably without retaining a copy thereof, and storage of the information encrypted with the one second rolling key, e.g. with the first private rolling key, and of the one second rolling key itself. The storage device is further configured to store the information encrypted with the one second rolling key, e.g. with the second private rolling key, and the one second rolling key itself, in accordance with the storage directive of the processor.

Beneficially, the processor is further configured with logic to generate the first asymmetric rolling crypto-key. If so, the processor also encrypts the information with the one first rolling key, e.g. the first private rolling key, and directs storage of this encrypted information and the one first rolling key itself. The storage device then stores the information encrypted with the one first rolling key, e.g. the first private rolling key, and the one first rolling key itself, in accordance with the storage directive.
According to an aspect of the invention, the information is a factor for generating a private key of a private-public key pair of an asymmetric crypto-key associated with the user. Here also, the user’s asymmetric crypto-key could be an RSA type asymmetric crypto-key, or another type asymmetric crypto-key.

If the information is such a factor, the processor may advantageously be further configured with logic to generate the private key of the user’s asymmetric crypto-key based on the decrypted factor. In a particularly preferred implementation, the private key of the user’s asymmetric crypto-key is a split private key having a first private key portion and a second private key portion. The factor is one of multiple factors for generating the first private key portion of the split private key. In such an implementation, the processor is further configured with logic to generate the first private key portion of the user’s asymmetric crypto-key based on the decrypted factor and another of the multiple factors corresponding to a password associated with the user. The processor can then direct transmission of the generated first private key portion to further authenticate the user.

According to another aspect of the invention, the processor is also configured with logic to direct transmission of proof of knowledge of the stored one second rolling key, e.g. the second private rolling key, to authenticate the user, and of a request for the other of the second private rolling key and the second public rolling key, e.g. the second public rolling key. This transmission could, for example, be made in response to a user input indicating a desire to again login, although this is not mandatory. In any event, the processor receives the requested other second rolling key, e.g. the second public rolling key, responsive to this directed transmission, and decrypts the stored information encrypted with the one second rolling key, e.g. the second private rolling key, by applying the received other second rolling key, e.g. the second public rolling key. After so decrypting the information, the processor generates a third asymmetric rolling crypto-key having a third private rolling key and a third public rolling key. The process then encrypts the information with one of the third private rolling key and the third public rolling key, and directs transmission of the other of the third private rolling key and the third public rolling key, e.g. the third public rolling key, via a network. Here again, a copy of the other third rolling key is preferably not retained by the processor. The processor also directs storage of the information encrypted with the one third rolling key, e.g. encrypted with the third private rolling key, and the one third rolling key itself. The storage device can be further configured to store the information encrypted with the one third rolling key and the one third rolling key itself, in accordance with the storage directive.

In one preferred embodiment of the invention, a first network station, associated with a first network entity, is configured with logic to transmit, via a network, proof of knowledge of one of a first private rolling key and a first public rolling key of a first asymmetric rolling key, and a request for the other of the first private rolling key and the first public rolling key. A second network station, associated with a second network entity, is configured with logic to authenticate the first network entity based on the transmitted proof of knowledge of the one first rolling key, and transmit, via the network, the other first rolling key based on the transmitted request and the authenticating of the first network entity. It should be understood that each of the network stations could be a network server, personal computer (PC) or another type network device capable of performing the required functions.

The first network station is further configured with logic to decrypt the information encrypted with the one first rolling key by application of the other first rolling key transmitted by the second network station. After so decrypting the information, the first station generates a second asymmetric rolling key having a second private rolling key and a second public rolling key, encrypts the information with one of the second private rolling key and the second public rolling key, and stores the information encrypted with the one second rolling key along with the one second rolling key itself. The first station also transmits the other of the second private rolling key and the second public rolling key, via the network. The second network station is further configured with logic to store the transmitted other second rolling key. The first network station is typically further configured with logic to transmit, via the network, proof of knowledge of the stored one second rolling key and a request for the other second rolling key. If so, the second network station is further configured with logic to authenticate the first network entity based on the transmitted proof of knowledge of the one second rolling key, and to transmit the stored other second rolling key via the network, based on the transmitted request for the key and the authenticating of the first network entity. The first network station can then decrypt the stored information encrypted with the one second rolling key by application of the transmitted other second rolling key. After so decrypting the information, the first network station can also generate a third asymmetric rolling key having a third private rolling key and a third public rolling key, encrypt the information with one of the third private rolling key and the third public rolling key, and store the information encrypted with the one third rolling key along with the one third rolling key itself. The first network station additionally transmits the other of the third private rolling key and the third public rolling key, via the network, and the second network station stores the transmitted other third rolling key.

Advantageously, the first network station is further configured with logic to generate the first asymmetric rolling key, encrypt the information with the one first rolling key, store the information encrypted with the one first rolling key along with the one first rolling key itself, and transmit the other first rolling key via the network. The second network station is then further configured with logic to store the transmitted other first rolling key. In such a case, the one first rolling key transmitted by the first network station, is the stored one first rolling key, and the other first rolling key transmitted by the second network station is the stored other first rolling key.

It may also be beneficial for the first network station to be further configured with logic to generate, for the first network entity, an asymmetric crypto-key having a private key and a public key, to split the private key into a first private key portion corresponding to multiple factors and a second private key portion, and to transmit the second private key portion via the network. If the first network station is so configured, the second network station is further configured with logic to store the transmitted second private key portion. In such an implementation, the information could be one of the multiple factors.

If so, the first network station is further configured with logic to generate the first private key portion of the first network entity’s asymmetric crypto-key based on the decrypted factor and another of the multiple factors corresponding to a password associated with the first network entity, and to transmit proof of knowledge of the generated first private key portion via the network. The second network station is further configured with logic to further authenticate the first network entity based on the transmitted proof of knowledge of the first private key portion and the stored second private key portion.
BRIEF DESCRIPTION OF DRAWINGS

FIG. 1 depicts an exemplary network, including networked devices associated with users, a sponsor, merchants, and an optional distinguished entity, in accordance with the present invention.

FIGS. 2 depicts a computer suitable for use by a user to access a network, in accordance with the invention.

FIG. 3 is an exemplary block diagram of components of the computer depicted in FIG. 2.

FIG. 4 depicts a server suitable for use by a sponsor, a merchant, and/or an optional distinguished entity, in accordance with the present invention.

FIG. 5 is an exemplary block diagram of components of the server depicted in FIG. 4.

FIGS. 6A-6C depict a flow chart showing operations performed during the key generation protocol for initially generating and distributing a multifactor split asymmetric crypto-key with a rolling key to enhance the security of a stored factor, in accordance with the present invention.

FIGS. 7A-7B depict a flow chart showing certain operations performed during the user log-in protocol, including generating and distributing a new rolling key to enhance the security of a stored factor, in accordance with the present invention.

FIG. 8 depicts a flow chart showing other operations performed during the user log-in protocol, including user authentication and completion of login requirements.

PREFERRED EMBODIMENT OF THE INVENTION

The Network Architecture

FIG. 1 illustrates a network 10, which could be the Internet or another public or private network. As shown, the network 10 includes multiple network devices interconnected so as to be capable of communicating with each other. These network devices include network devices 30-33 associated with individual users, network devices 40-41 associated with merchants, a network device 50 associated with a sponsor, and optional network devices 60-62 associated with entities known to and trusted by the sponsor. These latter entities are sometimes referred to as distinguished entities.

User network devices 30-33 are typically personal computers, but could be other types of network devices. Merchant network devices 40-41 could be associated with any type entity having a presence, e.g., a web page, on network 10, and typically take the form of network, e.g., Internet, servers. Accordingly, the merchant network devices 40-41 will sometimes be referred to as merchant servers. The sponsor network device 50 also typically takes the form of a network server, and will sometimes be referred to as a sponsor server or sponsor station. Likewise, the optional distinguished entity network devices 60-62 typically take the form of network servers. Accordingly, the optional distinguished entity network devices 60-62 will sometimes be referred to as distinguished servers. It will be understood that a network may consist of more network devices than depicted in FIG. 1.

The User Network Device

FIGS. 2 and 3 depict an exemplary personal computer (PC) suitable for use by an individual user as a user network device 30-33 to access the network 10. The PC is preferably a commercially available personal computer. It will be recognized that the PC configuration is exemplary, and that other components (not shown) could be added or substituted for those depicted and certain of the depicted components could be eliminated if desired. Further, a user device 30-33 could be another type device, i.e., a device other than a computer, such as, but not limited to, a personal digital assistant (PDA) or a mobile phone, e.g., cellular or other type wireless communications device.

The PC or other type user network device functions in accordance with stored programmed instructions, which are sometimes characterized as logic, that are executed, drive its operation. Preferably, the computer stores its programmed instructions on an EPROM, or hard disk. It will be recognized that only routine programming is required to implement the instructions required to drive the computer to operate in accordance with the invention. Further, since the computer components and configuration are well understood by those skilled in the art, routine operations performed by depicted components will generally not be described, such operations also being well understood in the art.

Referring to FIG. 2, the computer 200 includes a main unit 210 with slots 211, 212, and 213, respectively provided for loading programming or data from a floppy disk, compact disk (CD), or other removable media, onto the computer 200. The computer 200 also includes a keyboard 230 and mouse 240, which serve as user input devices. A display monitor 220 is also provided to visually communicate, i.e., display, information to the user.

As depicted in FIG. 3, the computer 200 has a main processor 300 which is interconnected via bus 310 with various remote or local storage devices which may include, but are not limited to, erasable programmable read only memory (EPROM) 322, read only memory (RAM) 323, hard drive 324, which has an associated hard disk 325, CD drive 326, which has an associated CD 327, floppy drive 328, which has an associated floppy disk 329, USB port 395 for connecting a universal serial bus (USB) drive 396 (often called a flash drive or memory), and smart card reader 397 for communicating with a smart card 398. Also shown in FIG. 3 is a trusted processing module (TPM) 399 for securely storing cryptographic keys. Taken together, the remote and local storage devices will be referred to collectively as storage devices 370.

A drive controller 350 controls the hard drive 324, CD drive 326 and floppy drive 328. Also depicted in FIG. 3 is a display controller 320 interconnected to display interface 321, a keyboard controller 330 interconnected to keyboard interface 331, a mouse controller 340 interconnected to mouse interface 341 and a modem 360 interconnected to I/O port 365, all of which are connected to the bus 310. The modem 360 and interconnected I/O port 365, serve as a network interface, and are used to transmit and receive signals via the network 10 as described herein. Those skilled in the art will recognize that another type of network interface could be easily substituted for that shown. It should also be understood that, if desired, other components could be connected to the bus 310, or that less than all the components shown in FIG. 3 could be connected to the bus 310. By accessing and executing the stored computer programming, i.e., the stored programmed instructions, the processor 300 is driven to operate in accordance with the present invention.

The Sponsor, Merchant and/or Distinguished Entity Network Device

As noted above, the sponsor network device 50, each merchant network device 40-41 and each optional distinguished network device 60-62 is typically represented on network 10 by a network server. However, here again, any network compatible device which is capable of functioning in the described manner could be utilized in lieu of a server to represent the associated entity, e.g., the sponsor, each merchant and/or each optional distinguished entity, on the network.
FIGS. 4 and 5 depict an exemplary network server suitable for use by the sponsor, merchants, and optional distinguished entities to access the network 10 in accordance with the invention. The server is preferably a commercially available, high power mini-computer or mainframe computer. Here again, it will be recognized that the server configuration is exemplary, and that other components (not shown) could be added or substituted for those depicted and certain of the depicted components could be eliminated if desired.

The server functions as described below in accordance with stored programmed instructions, i.e. logic, which drive its operation. Preferably, the server stores its unique programmed instructions on an EPROM or hard disk. It will be recognized that only routine programming is required to implement the instructions required to drive the server to operate in accordance with the invention. Further, since the server components and configuration are well understood to those skilled in the art, routine operations performed by depicted components will generally not be described, such operations also being well understood in the art.

Referring to FIG. 4, the server 400 includes a main unit 410 with slots 411, 412, 413 and 414, respectively provided for loading programming or data from a floppy disk, CD and/or hard disk onto the server 400. The server 400 also includes a keyboard 430 and mouse 440, which serve as user input devices. A display monitor 420 is also provided to visually communicate, i.e. display, information to the applicable entity.

As depicted in FIG. 5, the server 400 has a main processor 500 which is interconnected via bus 510 with various storage devices including EPROM 522, RAM 523, hard drive 524, which has an associated hard disk 525, CD drive 526, which has an associated CD 527, and floppy drive 528, which has an associated floppy disk 529. The memories, disks and CD all serve as storage media on which computer programming or data can be stored for access by the processor 500. The stored data includes one or more databases containing information associated with network users or other network entities. Taken together, the storage devices will be referred to collectively as storage devices 570.

A drive controller 550 controls the hard drive 524, CD drive 526 and floppy drive 528. Also depicted in FIG. 5 is a display controller 520 interconnected to display interface 521, a keyboard controller 530 interconnected to keyboard interface 530, a mouse controller 540 interconnected to mouse interface 541 and a modem 560 interconnected to I/O port 65, all of which are connected to the bus 510. The modem 560 and interconnected I/O port 565 serve as a network interface and are used to transmit and receive signals via the network 10 as described herein. Those skilled in the art will recognize that another type of network interface could be easily substituted for that shown. It should also be understood that, if desired, other components could be connected to the bus 510, or that less than all the components shown in FIG. 5 could be connected to the bus 510. By accessing the stored computer programming, the processor 500 is driven to operate in accordance with the present invention.

Multi Factor Split Private Key Asymmetric Cryptography

As described above in the background, in asymmetric cryptography, the asymmetric crypto-key has a private-public key pair, which will be referred to below as D and E, with ciphertext C=Encrypt(M,E) and Decrypt(C,D)=M, where M is a message encrypted by the entity, other than the entity associated with the asymmetric crypto-key, which has access to the associated entity’s public key E, and decrypted by the entity associated with the asymmetric crypto-key using that entity’s private key D. It will be recognized that reverse relationship also holds true. That is, the private key D could alternatively be used to encrypt the message M thereby creating the ciphertext, in which case the public key E is applied to decrypt the message M.

In split private key asymmetric cryptography, the private key is split into multiple private key portions, which will be referred to below as D1, D2, where D=D1*D2 mod D(N). It will be recognized that, if desired, the private key could split into more than two private key portions, e.g. D1, D2, and D3, where D=D1*D2*D3 mod D(N). Thus, with the split private key, ciphertext C=Encrypt(M,E) and Decrypt(C,D1,D2)=M.

In multifactor split private key asymmetric cryptography, at least one of the multiple private key portions, e.g. D1 and/or D2, is derived from multiple factors. The invention will be described below with respect to one private key portion of the asymmetric crypto-key, which will be referred to as the first private key portion D1, being derived using the multiple factors. Additionally, while the invention will be described below with respect to the first private key portion D1 being derived from two factors, it should be understood that the invention can be easily adapted to a private key portion derived from more than two factors, as will be well understood by those skilled in the art from the following description.

The entity associated with the first private key portion D1 will be referred to below as a user. However, it should be understood that preferably each individual network and each distinguished entity represented on the network 10, is associated with a multifactor split asymmetric crypto-key and hence has an associated first private key portion. Additionally, if desired, each merchant and sponsor represented on the network 10 could also be associated with a multifactor split asymmetric crypto-key and, if so, would likewise have an associated first private key portion.

The user is represented on the network by a user network device, such as a PC, as has been described above. As discussed above in the background, each of the multiple factors will be under the control of the associated user or other entity. For purposes of the description below, one of the factors will correspond to the user’s password and be sometimes referred to as Fp. Preferably, Fp is derived from the user’s password and not the user’s password itself, although this is not mandatory, an Fp could, if desired, be the user’s password. The other factor will preferably be a crypto-key and be sometimes referred to as Fk. If the implementation included three factors, each of the other two factors would be a different crypto-key and might be sometimes referred to as Fk1 and Fk2.

It should be recognized that the use of factors corresponding to the user’s password and to a crypto-key(s) is not mandatory, and those skilled in the art will understand that other type factors could be used if so desired. Additionally, although derivation of the first private key portion D1 could, if desired, be accomplished by simply concatenating the multiple factors, preferably the first private key portion D1 is derived by cryptographically combining Fp and Fk, or, in a three factor private key portion, Fp, Fk1 and Fk2, using the PKCS-5 algorithm, as will be discussed further below.

The first private key portion D1 is not stored in a persistent state, but must be generated whenever its use is desired. The first private key portion is however temporarily stored after being generated, so as to be usable to prove the user’s identity multiple times without the user providing any authentication information. This temporary storage may be limited to a predefined time period, or to a predefined number of authentications.

The other private key portion, which will be referred to as the second private key portion D2, is preferably stored in a
persistent state under control of a network entity other than the user. The public key \( E \) is also preferably stored in a persistent state under the control of other than the user, so as to be available to at least one other entity. In some implementations, the public key \( E \) may be beneficially stored so as to be available to each and every merchant, sponsor distinguished entity, and/or other user represented on the network 100. The second private key portion \( D_p \) and public key \( E \) could be stored under the control of a sponsor, a merchant or a distinguished entity, or the second private key portion \( D_p \) and public key \( E \) could be respectively stored under the control of a different one of a sponsor, a merchant and a distinguished entity.

In some implementations, the second private key portion \( D_p \) and public key \( E \), or some portion of the public key \( E \), are retained by the sponsor station 50, e.g. on hard disk 525, in association with a user's identifier (ID). In these implementations it may also be beneficial for the public key \( E \) to be stored on each merchant server 40-41, e.g. on hard disk 525, or on each merchant server 40-41 and each user PC 30-33, e.g. on hard disk 325, in association with a user's identifier (ID).

Storage Of The Factors

As described in the background, preferably neither the user's password nor the corresponding factor \( F_p \) is stored, except temporarily on RAM 323 of the user's PC 200 for only the time necessary, in the case of the password, to allow for generation of the corresponding factor \( F_p \) after the user enters the password and, in the case of the corresponding factor \( F_p \), to allow for generation of the first private key portion \( D_p \) after the corresponding factor \( F_p \) has been generated.

On the other hand, the other factor \( F_p \) may be stored on either a user's PC 200, such as on hard disk 325, or on removable media configured to communicate with the user's PC, such as on memory stick 396, which is shown as USB flash memory, smart card 398, floppy disk 329, compact disk (CD) 327, or even as a zip drive (not shown) or digital camera (not shown). In a three factor implementation, beneficially the factor \( F_p \) is stored on the hard disk 325 of the user's PC 200, while the factor \( F_p \) is stored on a memory stick 396.

Securing A Stored Factor of A Multifactor Split Asymmetric Crypto-Key

As described in the above referenced application, in the multifactor split private key asymmetric cryptosystem developed by Tricipher, Inc., the factor \( F_p \) in a two factor implementation, or each of the factors \( F_{k1} \) and \( F_{k2} \) in a three factor implementation, is preferably the private key of a private-public asymmetric key pair, including \( D_{k1} \), \( D_{k1} \), or \( E_{k1} \), \( E_{k2} \), \( D_{k2} \), or \( E_{k2} \), as applicable, and having a modulus \( N_{F_{k1}} \) or \( N_{F_{k2}} \), as applicable. For example, \( D_{k1} \) and \( E_{k1} \), as well as \( D_{k2} \) and \( E_{k2} \), may each form a conventional RSA private-public asymmetric key pair, although this is not mandatory. If the factor is stored on a user's computing device it is sometimes referred to as \( D_{USB} \). On the other hand, if the factor is stored on removable media it is sometimes referred to as \( D_{USB} \).

In a preferred three factor implementation of the multifactor split private key asymmetric cryptosystem developed by Tricipher, Inc., \( F_{k1} = E_{other} \) is stored on a user's computing device and is the private key of a private-public asymmetric key pair, including \( D_{other} \) and \( E_{other} \), and having a modulus \( N_{other} \) and \( E_{k2} = D_{other} \) is stored on removable media and is the private key of a private-public asymmetric key pair, including \( D_{USB} \) and \( E_{USB} \), and having a modulus \( N_{USB} \). The non-private parts of the factor keys, i.e., \( E_{other} \), \( N_{other} \), \( E_{USB} \), and \( N_{USB} \), are stored, for example, as a merchant, sponsor or distinguished entity server.

Typically, \( D_{other} \) is stored securely on the hard disk 324 of a user's PC 200 using the protection capabilities provided by the PC's operating system, preferably as a non-exportable private key in a Windows Operating System key-store. Of course, as desired, \( D_{other} \) could be stored in a Windows Operating System registry. Alternatively, \( D_{other} \) can be, as desired, stored in the trusted processing module (TPM). No matter where or how on the user's computing device \( D_{other} \) is stored, in the most basic configuration, \( D_{other} \) can only be used from the user's computing device upon which it is stored. That is, \( D_{other} \) is a non-exportable private key, stored only on the user device at which it will be used. However, as detailed in the above referenced application, the multifactor split private key asymmetric cryptosystem developed by Tricipher, Inc. also facilitates porting \( D_{other} \) to other devices for use thereat.

\( D_{USB} \) is stored on removable media configured to communicate with the user's PC, perhaps most commonly on a USB flash memory 396. \( D_{USB} \) also needs to be protected, since storing any kind of key in the clear should be avoided if possible. In the case of \( D_{USB} \) this is particularly important because, if \( D_{USB} \) is stored on removable media in clear and the user misplaces or otherwise loses the removable media, an attacker could easily access, extract and/or copy \( D_{USB} \) from the removable media, and potentially use \( D_{USB} \) to impersonate the user. Thus, in the multifactor split private key asymmetric cryptosystem developed by Tricipher Inc., \( D_{USB} \) is beneficially stored on the removable media in an encrypted state, as also indicated in the above referenced application.

Preferably, encryption of \( D_{USB} \) is not performed with the user's password, because this would still leave \( D_{USB} \) vulnerable to dictionary attacks and guessing attacks. If the factors include both \( D_{other} \) and \( D_{USB} \), as disclosed in the above referenced application, be encrypted using \( D_{other} \), which is stored, for example, on the user's PC. However, in a two factor implementation, where the factors include \( D_{USB} \), among others, there is no treading key to encrypt \( D_{USB} \). Furthermore, even in a three factor implementation, if \( D_{USB} \) is encrypted using \( D_{other} \) an attacker with access to the removable media, as well as the user's PC or other computing device storing \( D_{other} \) and the user's password, could obtain \( D_{USB} \) from the user's computing device and decrypt \( D_{USB} \) by applying \( D_{other} \). This would in turn allow the attacker to use the user's password, \( D_{USB} \) and \( D_{other} \), to generate the first private key portion, e.g. D1. Having generated the first private key portion, the attacker would now be in a position to impersonate the user.

Accordingly, the following describes a technique to further improve the security of stored factors, such as \( D_{USB} \) and/or \( D_{other} \), which will be used to generate multifactor split private keys of an asymmetric crypto-key.

Key Generation Protocol—Initial Generation and Distribution of A Multifactor Split Asymmetric Crypto-Key with A Rolling Crypto-Key to Enhance the Security of a Stored Factor

Referring now to FIG. 6, preferably the generation of the multifactor split asymmetric crypto-key begins with the generation of the factors. The following operations will be discussed in terms of a two factor implementation. In this implementation, one factor corresponds to the password, which has sometimes been referred to as \( F_p \). The other factor is the private key of an asymmetric crypto-key generated for storage on removable media, which has sometimes been referred to above as \( F_k \).

-Generation of the Factor-

In this implementation, the processor 300 of the user's PC 200, executing programmed instructions stored on the hard disk 325, generates a first asymmetric crypto-key pair in step 600. As the factor will, in the particular implementation being
described, be stored on the memory stick 396, which is shown as USB flash memory, the asymmetric crypto-key pair generated in step 600 will, hereafter, be referred hereafter as $D_{USB}, N_{USB}$ and $E_{USB}, N_{USB}$, where $D_{USB} \leftarrow F_{R}$. Preferably, the crypto-key pair is an RSA asymmetric crypto-key, although this is not mandatory and another type of crypto-key or even something other than a crypto-key could be used, if desired. In other implementations, this factor could, if desired, be generated by the processor 500 of a network server 400, executing programmed instructions stored on the hard disk 525. As $N$ is a well known modulus and not of particular relevance to the present invention, the generated asymmetric crypto-key pair will, at times, be described below only in terms of the private key $D_{USB}$ and public key $E_{USB}$.

-Generation of the Rolling Key-

At step 605, the processor 300 of the user’s PC 200 also generates another, or second, asymmetric crypto-key pair, having a private key $D_{R}$ and a public key $E_{R}$, and a modulus $N_{R}$. This asymmetric crypto-key pair will serve as a rolling crypto-key and be used to secure $D_{USB}$ when stored on the memory stick 396. The processor 300 associates this rolling key with $D_{USB}, E_{USB}, N_{USB}$ and, preferably, the rolling key is an RSA asymmetric crypto-key pair, although this is not mandatory as another type of asymmetric crypto-key pair could also be used as the rolling key. Here, again, in other implementations, the rolling key could, if desired, be generated and associated with the factor $D_{USB}$ by the processor 500 of a network server 400, executing programmed instructions stored on the hard disk 525.

-Securing the Factor with the Rolling Key-

In step 610, the processor 300 of the user’s PC 200 encrypts the factor $D_{USB}$ with either the private key $D_{R}$ or public key $E_{R}$ of the rolling asymmetric key pair generated in step 605. In this implementation, private key $D_{R}$ is used for this encryption, i.e. ciphertext $CD_{USB}=E_{CD_{USB}, D_{R}}$. Thus, $D_{USB}$ can now only be used to derive the first private key portion if it is first decrypted using public key $E_{R}$, i.e. $E_{CD_{USB}} E_{R}=D_{USB}$.

Optionally, but preferably, at step 612 the processor 300 of the user’s PC 200 encrypts $D_{R}$, e.g. ciphertext $CD_{D_{R}}=E_{CD_{D_{R}}, D_{R}}$. In this regard, one option is to encrypt $D_{R}$ using a key which is stored on the user’s PC 200, such as another of the multi-factors for generating $D_{R}$, e.g. $D_{other}$, in which case the key would typically be different for each user. Another option is to encrypt $D_{R}$ using a key that is built into the operating system or other software that is being executed by processor 300 of the user’s PC 200, in which case the key would typically be the same for each user. It will be recognized that the key used to encrypt $D_{R}$ could be an asymmetric or symmetric key, but the rolling key $D_{R}$ itself must be part of an asymmetric crypto-key pair, because one of the keys of the asymmetric private-public crypto-key pair, in this implementation, must be stored by an entity other than the user.

At step 615, the processor 300 of the user’s PC 200 stores the optionally encrypted private rolling key $D_{R}$ in association with, e.g. co-located with, ciphertext $CD_{USB}$ on the memory stick 396.

-Distribution of the Rolling Key-

At step 620, the processor 300 of the user’s PC 200 transmits the non-private portions of the generated crypto-key pair, i.e., $E_{USB}$ and $N_{USB}$, as well as the other key of the rolling crypto-key pair not used to secure, i.e. encrypt, the factor $D_{USB}$, to a second network device. Since the first network device is associated with the user in the implementation being described, the second network device is associated with a network entity other than that user. The entity could be a sponsor, merchant or distinguished entity represented on the network by a server as discussed above. In this implementation the transmission is to sponsor station 50 via network 10. Since private rolling key $D_{R}$ is used to secure the factor $D_{USB}$, public rolling key $E_{R}$ and $N_{R}$ are directed by the processor 500 to sponsor station 50 via network 10. In accordance with the processor directive, the modem 360 transmits $E_{USB}, N_{USB}$ and $E_{R}, N_{R}$ via the I/O port 365. It should be appreciated that this transmission could be broken into multiple transmissions, and/or could be encrypted with the symmetric key/one time activation code, or even another key. Preferably, at least the transmitted $E_{USB}$ and $E_{R}$ are encrypted with the symmetric key/one time activation code, or other key. Of course, as desired, either or both of $N_{USB}$ and $N_{R}$ could also be encrypted. The sponsor station server 400 receives $E_{USB}, N_{USB}$ and $E_{R}, N_{R}$ via the I/O port 565. The received $E_{USB}, N_{USB}$ and $E_{R}, N_{R}$ are directed by the modem 560 to the processor 500.

Optionally, but preferably, at step 622 the processor 500 of the sponsor server 400 encrypts $E_{R}$, i.e. ciphertext $CE_{R}=E_{R}, K_{U}$ and $E_{USB}$, i.e. ciphertext $CE_{USB}=E_{USB}, K_{USB}$ with regard to $E_{R}$, one option is to encrypt $E_{USB}$ using a key that has been generated and stored on the sponsor server 400, e.g. a generated symmetric key. Another option is to encrypt $E_{USB}$ using a key that is built into the operating system or other software that is being executed by processor 500 of the sponsor’s server 400. With regard to $E_{USB}$, one option is to use the public key $E_{R}$ to encrypt $E_{USB}$. Other options are to encrypt $E_{USB}$ using a key, e.g. a symmetric key, that has been generated and stored on the sponsor’s server 400 or a key that is built into the operating system or other software that is being executed by processor 500 of the sponsor’s server 400. It will be recognized that the key(s) used to encrypt $E_{USB}$ and/or $E_{USB}$ can be asymmetric or symmetric, but the rolling key $E_{R}$ itself must be part of an asymmetric crypto-key pair, because one of the keys of the asymmetric private-public crypto-key pair, in this implementation, must be used to secure the factor $D_{USB}$ and be stored by the user device.

At step 625, the processor 500, in turn, directs the storage of received, optionally encrypted $E_{USB}$ and $N_{USB}$ in association with received, optionally encrypted $E_{R}$ and $N_{R}$ preferably on the hard disk 525 of the sponsor station server 400. Whether or not encrypted, $E_{USB}, N_{USB}$ and $E_{R}, N_{R}$ are preferably stored in a persistent state on hard disk 525, in association with an applicable user’s identifier (ID).

At step 630, after directing transmission of $E_{USB}, N_{USB}$ and $E_{R}, N_{R}$ at step 620, the processor 300 of the user’s PC 200 destroys at least $E_{USB}$ and $E_{R}$. The processor 300 of the user’s PC 200 may also, if desired, destroy $N_{USB}$ and $N_{R}$. Alternatively, the processor no longer has access to $E_{USB}$, and $E_{R}$, and additionally may also no longer have access to $N_{USB}$ and $N_{R}$.

Accordingly, after steps 625 and 630 are completed, the applicable user device 30 has access to the ciphertext $CD_{USB}$ which is the factor $D_{USB}$ encrypted with the rolling key $D_{R}$ and the rolling key $D_{R}$ itself stored on the user’s USB memory stick 396, whenever the portable memory stick 396 is interconnected with the applicable user device 30. The sponsor station 50 has access to public factor key $F_{USB}, N_{USB}$ and public rolling key $E_{R}, N_{R}$ which are persistently stored in association with the user’s ID on hard disk 525.

-Generation of the User’s Multi-factor Split Asymmetric Crypto-Key-

At step 635, the processor 300 of the user’s PC 200 generates yet another, i.e. a third, asymmetric crypto-key pair, including private key $D_{ch}$, public key $E_{ch}$, and modulus $N_{ch}$, which will serve as the user’s asymmetric crypto-key. Here again, in other implementations, the user’s asymmetric
crypto-key could, if desired, be generated by the processor 500 of a network server 400, executing programmed instructions stored on the hard disk 525.

At step 640, the processor 300 of the user’s PC 200 splits Dz into at least a first private key portion Dz₁, which corresponds to the first private key portion sometimes referred to as D₁, and a second private key portion Dz₂, which corresponds to the second private key portion sometimes referred to as D₂, by first determining Dz₁/3, and then determining Dz₂ utilizing conventional techniques based on the well known relationship between Dz₁ and Dz₂, i.e. Dz₁ * Dz₂ = Dmod(N) (N).

The processing to determine Dz₁ is based upon the multiple factors. As described above, in this implementation the multiple factors are the factor Fₚ, which corresponds to the user’s password, and the factor Dz₃, which is a private key of an asymmetric crypto-key pair. However, as noted above, the multiple factors could be any two or more of Fₚ, Dz₃, another factor commonly referred to as Dz₄, which is derived in a manner similar to Dz₃ but stored on the user’s PC 200 and/or other type factors.

In step 645, the processor 300 of the user’s PC 200 calculates Dz₄ utilizing the PKCS-5 algorithm, which is a well known one way function preferably stored on the hard disk 325. The preferred equation for determining Dz₄ is as follows:

\[ D_{z4}^{PKCS-5}(\text{sign} \{ \text{Sha-1(password), Dz}_3, \text{salt, iteration count} \}) \]

\[ (1) \]

Thus, as shown in equation (1), Dz₄ is computed in by taking a first Sha-1 hash of the password, where Sha-1 (password) = Fₚ, and transforming this quantity with Dz₃ to create Dz₄ASP, and then using the result of this transformation as an input to the PKCS-5 algorithm, along with the salt and the iteration count. After the determination of Dz₄, the user’s PC 200 has knowledge of Dz₁, Dz₂, and Dz₄. Accordingly, in step 650, the processor 300 of the user’s PC 200 can determine Dz₂ based on the relationship Dz₁*Dz₂ = Dmod(N), and thereby complete the splitting of Dz₁.

For those cases in which it is desired to use the password, Dz₃, and Dz₄ASP as the multiple factors, the following equation for determining Dz₄ is used:

\[ D_{z4}^{PKCS-5}(\text{sign} \{ \text{Sha-1(password), Dz}_3, \text{Dz}_4^{ASP}, \text{salt, iteration count} \}) \]

\[ (2) \]

Thus, as shown in equation (2), Dz₄ is computed by taking a first Sha-1 hash of the password, where Sha-1 (password) = Fₚ, and transforming this quantity with Dz₄ASP, taking a second Sha-1 hash of the transformed quantity and transforming the second Sha-1 hash of the transformed quantity and transforming the second Sha-1 hash of the transformed quantity with Dz₃ASP and then using the result of this second transformation as an input to the PKCS-5 algorithm, along with the salt and the iteration count. After the determination of Dz₄, the user’s PC 200 has knowledge of Dz₁, Dz₂, and Dz₄. Accordingly, the processor 300 of the user’s PC 200 can determine Dz₂ based on the relationship Dz₁*Dz₂ = Dmod(N), and thereby complete the splitting of Dz₁.

For those cases in which it is desired to use the password and Dz₄ASP as the multiple factors, but not Dz₃, the following equation for determining Dz₄ is can be used:

\[ D_{z4}^{PKCS-5}(\text{sign} \{ \text{Sha-1(password), Dz}_4^{ASP}, \text{salt, iteration count} \}) \]

\[ (3) \]

Thus, as shown in equation (3), Dz₄ is computed by taking a first Sha-1 hash of the password, where Sha-1 (password) = Fₚ, and transforming this quantity with Dz₄ASP and then using the result of this transformation as an input to the PKCS-5 algorithm, along with the salt and the iteration count. After the determination of Dz₄, the user’s PC 200 has knowledge of Dz₁, Dz₄, and Dz₄ASP. Accordingly, the processor 300 of the user’s PC 200 can determine Dz₁ based on the relationship Dz₁*Dz₂ = Dmod(N), and thereby complete the splitting of Dz₂.

After the processor 300 of the user’s PC 200 determines Dz₂ at step 655, the processor 300 destroys Dz₁ and Dz₄.

-Distribution of the User’s Multifactor Split Asymmetric Crypto-Key-

At step 660, the processor 300 directs transmission of Dz₂, E₃, and N₃ to the entity with which the user is establishing a multifactor asymmetric crypto-key. As discussed above, the entity could be a sponsor, merchant or distinguished entity represented on the network by a server. However, in this particular implementation the transmission is to sponsor station 50 via network 10. In accordance with the processor directive, the modem 360 transmits Dz₂, E₃, and N₃ via the I/O port 365. It should be appreciated that this transmission could be broken into multiple transmissions, and could be encrypted with the symmetric key/time activation code, or even another key. Preferably, at least the transmitted Dz₂ is encrypted with the symmetric key/time activation code, or other key. Of course, as desired, either or both of E₃, and N₃ could also be encrypted. It will also be recognized that the transmission in steps 620 and 660 could be combined in a single transmission, if so desired.

The sponsor station server 400 receives Dz₂, E₃, and N₃ via the I/O port 365. The received Dz₂, E₃, and N₃ are directed by the modem 560 to the processor 500. Optionally, but preferably, at step 662 the processor 500 of the sponsor server 400 encrypts Dz₂, i.e. ciphertext CD₆₋₂⁻Encrypt(DCD₂₋₂, KCD₂₋₂), and E₃, i.e. ciphertext CEx₋₃⁻Encrypt(Ex₃, KEx₃). One option is to encrypt Dz₂ and/or E₃, using a key that has been generated by and stored by the sponsor server 400. Another option is to encrypt Dz₄ and/or E₃, using a key that is built into the operating system or other software that is being executed by processor 500 of the sponsor server 400. It will be recognized that the key(s) used to encrypt Dz₂ and/or E₃ could be asymmetric or symmetric.

At step 665, the processor 500 directs the storage of optionally encrypted Dz₂ and E₃, along with N₃, in association with E₃ASP, N₃ASP, and previously on the hard disk 525 of the sponsor station server 400. Dz₂, E₃, and N₃ are also preferably stored in a persistent state on hard disk 525, in association with the applicable user’s identifier (ID).

At step 667, after directing transmission of Dz₂, E₃, and N₃ at step 660, the processor 300 of the user’s PC 200 destroys Dz₂. The processor 300 may also destroy E₃, and N₃, if so desired. Accordingly, the processor no longer has access to Dz₂, and additionally may no longer have access to E₃ and N₃.

Alternatively the key association entity, in this implementation, sponsor station 50, could generate one or more of the first, second, or third asymmetric crypto-key pairs, and perform the splitting of Dz₂. If a device representing an entity other than the applicable user, e.g. sponsor server 400, generates either the first or the second asymmetric crypto-key pair, or both, it will direct transmission of one part of each key pair, i.e. the private or public key of the applicable generated asymmetric crypto-key pair, to the user’s PC 200, and destroy the copy at the server 400, so that only the user’s PC 200, through the storage on the hard disk 325 or USB memory stick 396, has access to the key or keys, in this implementation access to Dz₂ and/or Dz₃ASP. If a device representing an entity other than the applicable user generates the third asymmetric crypto-key pair and performs the splitting, it need not provide any associated information to the user’s PC 200. However, preferably the user will provide
the password to the generating device for use with the PKCS-5 algorithm. Also, the generating device having generated $D_{U1}$ and $D_{U2}$ destroys $D_{U3}$, $D_u$, and the password, and stores $D_{U2}$, preferably on a hard disk and in a persistent state.

Login Protocol—With Generation and Distribution of A New Rolling Key to Enhance the Security of A Stored Factor

The entity to which the user device 30 directed the transmissions described in steps 620 and 660, in this implementation a sponsor represented by the sponsor station 50, provides trusted authentication of an individual user during login. For this implementation, the login protocol is described in detail below in the context of a request for authentication being received by the user device 30 from the sponsor station 50. However, those skilled in the art will recognize that the described protocol is easily adapted to requests for authentication received from other than the sponsor station 50. It will also be understood that in the described protocol, the authentication request from the sponsor station 50 may represent an authentication request initiated by an entity other than the sponsor. That is, the authentication request may have been initiated by a merchant represented by a merchant server 40-41 or another user represented by another user device 31-33 or a trusted distinguished entity represented by distinguished server 60-62. Furthermore, if the user device 30 had directed the transmissions described in steps 620 and 660 to an entity other than the sponsor station 50, e.g. to a merchant represented by a merchant server 40-41 or another user represent by another user device 31-33 or a trusted distinguished entity represented by distinguished server 60-62, the authentication request could represent an authentication request initiated by a sponsor represented by sponsor station 50. It will also be recognized that a merchant server, such as merchant server 40, or a distinguished server, such as distinguished server 60, might also function as the sponsor station 50, although generally this will not be the case.

-Initial User Authentication and Unlocking the Secured Factor-

Referring now to FIG. 7, at step 700, a communication session is established between the user device 30 and the sponsor station 50 via network 10, and at step 705, the processor 500 of the sponsor’s station 50 directs transmission of and, in accordance with the directive, the modem 560 and I/O port 565 transmit a request, via network 10, to user device 30, requesting that the applicable user authenticate himself or herself to the sponsor station 50. The I/O port 565 receives the transmission and the modem 360 directs the request to the processor 300 of the user’s device 30. At step 710, the processor 300 of user device 30 directs the modem 360 and I/O port 565 transmit the user’s ID to the sponsor station 50 via network 10. It should be understood that while the user’s ID itself is not encrypted by the user’s device 30 and therefore can be considered as transmitted in the clear, in some, if not most, implementations communications between the user device 30 and the sponsor station 50 are preferably protected using server side SSL encryption, which is well understood by those skilled in the art and therefore not further described herein.

The transmitted user ID is received by I/O port 565 and directed by modem 560 to the processor 500 of the sponsor station 50. Next, at step 715, the processor 500 of sponsor station 50 generates a challenge, which consists of a random number R1 and a time stamp, and directs transmission of the same to the user device 30 via the network 10, also in the clear. In accordance with the directive the challenge is transmitted via the modem 560 and I/O port 565.

The transmitted challenge is received by the I/O port 365 and directed by the modem 360 to the processor 300. In response to the challenge, the processor 300 of the user device 30, at step 720, generates a request including a permission request (PR) that consists of the challenge, a time stamp, and a random number R2. The generated request also includes a request for $E_{U2SR}$, $N_{U2SR}$ and $E_{ER}$, $N_E$.

In order to maintain the user himself for herself to the sponsor station 50, the user, represented by the user device 30, must demonstrate knowledge of $D_{U1}$, the factors associated with $D_{U1}$, the factors associated with $F_u$, and/or the one of the private-public rolling crypto-key pair which has been used to secure a factor, in this implementation the private rolling key $D_{R}$ that has been used to secure factor $D_{U2SR}$. Preferably, the user must demonstrate knowledge of all these items. It should be noted that, as discussed above, the user device 30 does not provide the user’s password to the sponsor station 50. Thus, at step 720, the user makes the removable media 960 upon which $D_{U2SR}$ encrypted with $D_R$ and $D_{U2SR}$ itself are stored available to user device 30 for retrieval by processor 300.

If $D_{R}$ is stored in an encrypted state, in step 722 the processor 300 of user device 30 decrypts $D_{R}$, i.e., $D_{R} = \text{Decrypt}(C_{D_{R}}, K_{ER})$, prior to directing transmission of proof of knowledge of $D_{R}$, for example by transmitting $D_{R}$ itself or preferably a time stamp or challenge signed with $D_{R}$ to the sponsor station 50. In step 725, the processor 300 then directs and, in accordance with this directive, the modem 360 and I/O port 365 function to transmit the proof of knowledge of $D_{R}$ and the user’s ID, together with the generated request, which includes the PR and the request for $E_{U2SR}$, $N_{U2SR}$ and $E_{ER}$, $N_E$, to the sponsor station 50 via the network 10.

The information transmitted in step 725 is received by I/O port 565 and directed by modem 560 to processor 500. At step 730, the processor 500 of sponsor station 50 determines whether or not the proof of knowledge of $D_{R}$ transmitted by the user device 30 does in fact prove knowledge of the private rolling key that has been used to secure, i.e. encrypt, $D_{U2SR}$. This can be done using any of several well known techniques. For example, the processor 500 could encrypt a test message TM using the public rolling key $E_{ER}$, i.e. ciphertext CTM = Encrypt(TM, $E_{ER}$), which as discussed above is stored at the sponsor station 50, and then attempt to decrypt TM using the transmitted private rolling key $D_{R}$, i.e. $D_{R} = \text{Decrypt}(C_{D_{R}}, K_{ER})$. However, preferably the processor 500 need only recover a time stamp or challenge signed with $D_R$ by applying $E_{ER}$. If TM, or alternatively the time stamp or challenge, is successfully recovered, the user device 30 has proved to the sponsor station 50 that the user has knowledge of one of the private-public rolling crypto-key pair, i.e. in this implementation $D_{R}$, used to secure the factor $D_{U2SR}$. If not, the login will not be completed and the processor 500 preferably suspends the user’s ID.

If the processor 500 determines that the user has proved knowledge of the rolling crypto-key used to secure the factor $D_{U2SR}$ at step 735 the processor 500 retrieves $E_{U2SR}$, $N_{U2SR}$ and $E_{ER}$, $N_E$ from storage. If either or both $E_{U2SR}$ and $E_{ER}$ are stored in an encrypted state, in step 737 the processor 500 decrypts $E_{U2SR}$ and/or $E_{ER}$, i.e., $E_{U2SR} = \text{Decrypt}(C_{E_{U2SR}}, K_{ER})$ and/or $E_{ER} = \text{Decrypt}(C_{E_{ER}}, K_{ER})$, prior to transmitting these keys to the user station 30. At step 740, the processor 500 directs and, in accordance with this directive the modem 560 and I/O port 565 function to transmit $E_{U2SR}$, $N_{U2SR}$ and $E_{ER}$, $N_E$ to user device 30 via the network 10. In step 745, which preferably is performed at this point, the processor 500 destroys $E_{ER}$, $N_E$ or sets a flag or other indicator that $E_{ER}$, $N_E$ must be reset, or both.

The keys transmitted in step 740 are received by the I/O port 365 and directed by the modem 360 to the processor 300. At step 750, the processor 300 of user device 30 applies the
public rolling key $E_R$ to unlock the factor $D_{USR}$ that had been stored on the removable media 396 encrypted with the rolling private key $D_{R}$, i.e. ciphertext $CD_{USR} = \text{Encrypt}(D_{USR} \times D_R)$. More particularly, the processor 300 proceeds to decrypt ciphertext $CD_{USR}$ using the received public rolling key $E_R$, i.e. $\text{Decrypt}(CD_{USR} \times E_R) = D_{USR}$. In step 755, the processor 300 determines if $D_{USR}$ is successfully recovered. If so, the sponsor station 50 has proved to the user device 30 that the sponsor has knowledge of the other of the private-public rolling crypto-key pair that was not used to secure the factor $D_{USR}$, and hence authenticates the sponsor to the user. If not, the processor 300 could, for example, suspend use of the user's ID and notify the sponsor station 50 and perhaps other network entities of the suspension of the user's ID.

-Resetting the Rolling Key-

After the processor 300 successfully decrypts the factor $D_{USR}$ in step 760, processor 300 preferably destroys $D_R$ so that it is no longer available or stored on the removable media 396. In step 765, the user device 30 and sponsor station 50 reset the rolling asymmetric crypto-key. In this regard, processor 300 generates a new rolling asymmetric crypto-key pair having a private key, which will be referred to as $E_{R1}$, a public key, which will be referred to as $F_{R1}$, and a modulus $N_{R1}$. In this implementation the processor 300 of user device 30 again secures the factor, this time by encrypting $D_{USR}$ with $D_{R1}$, and stores the secured factor, i.e. $D_{USR}$ encrypted with $D_{R1}$, along with $D_{R1}$ itself, on the removable media 396. The processor 300 also directs and, in accordance with this directive the modern 360 and I/O port 365 function to transmit $E_{R1}$, $F_{R1}$, to the sponsor station 50, which stores $E_{R1}$, $N_{R1}$, in association with $E_{USR}$, $N_{USR}$ to reset $E_R$, $N_R$, which are deleted from the hard disk 525 storage of sponsor station server 400. After distribution to the sponsor station 50, the user device 30 destroys its copy of $E_{R1}$, $N_{R1}$. The above described resetting is performed at each new login and in the manner described in steps 605-625 with reference to $E_{R1}$, $N_{R1}$, as applicable.

-Deriving the User's Portion of the Split Private Key and Completing Initial Login Requirements-

At step 770, the processor 300 of the user device 30 directs the display controller 520 to present, on display interface 521, a request that the user enter the user's password. In response to the request, the user enters a password via keyboard interface 331, which is directed by keyboard controller 320 to processor 300. After receipt of the user's password, at step 775 the processor 300 of user device 30 computes $D_{L1}$, by taking the first Sha-1 hash of the user's password, where Sha-1 (password) = $F_p$, and then transforming this quantity with recovered factor $D_{USR}$, i.e. ciphertext $CF_{p} = \text{Encrypt}(F_p, D_{USR})$, and then using the result of this transformation, i.e. $CF_{p}$, as an input to the PKCS-5 algorithm, along with the salt and the iteration count. Thus, after completing step 775, the initial login procedure is completed and the user device 30 has knowledge of the first private key portion $D_{L0}$ of the split private key $D_L$ of the user's asymmetric crypto-key.

-Final User Authentication and Completing the Login Requirements-

Because in this implementation only two factors are used to generate the private key portion $D_{L1}$, at step 800 the processor 300 of user network device 30 creates a signed permission request (SPR) using only $D_{L1}$ and $D_{USR}$, i.e. ciphertext $\text{CSRP} = \text{Encrypt}(\text{sign}(PR, D_{USR}), D_{L1})$. Note that neither the rolling key nor the password related factor $F_p$ are applied. It should, however, be noted that while the factor corresponding to the password, i.e. $F_p$, is preferably not used, it could be used in the SPR if so desired. The signed permission request has the following form:

$$\text{CSRP=\text{Enc}}(\text{sign(Sha-1(\text{PR}), D_{USR}), D_{L1}})$$

Thus, in this implementation PR is combined with a Sha-1 hash of PR signed with $D_{USR}$, and this combination is then encrypted with $D_{L1}$. If an additional factor(s) were used to generate the split private key portion $D_L$, in the present implementation, Sha-1 hash of PR signed with each additional factor and encrypted with $D_{L1}$ would also preferably be included in the a signed permission request (SPR). At step 805, the processor 300 of user device 30 directs and, in accordance with this directive the modem 360 and I/O port 365 function to transmit CSRF to the sponsor station 50 via network 10.

The I/O port 565 receives the transmitted CSRF and modem 560 directs the received CSRF to processor 500. In step 810, the processor of sponsor station 50 decrypts if necessary and applies $D_{USR}$, $E_R$, and $N_R$, which as discussed above have been previously stored by the sponsor station 50, to decrypt and thereby recover PR and sign (Sha-1(PL), $D_{USR}$), i.e. $\text{Decrypt}$(CSRP, $D_{USR}$), $E_R$, $N_R$, for $\text{sign}$(Sha-1(PL), $D_{USR}$). In step 815, the processor 500 verifies that the challenge included in the recovered PR is correct and, if so, thereby explicitly verifies the user's knowledge of the factor $D_{L1}$. If not, the sponsor station processor 500 may suspend the user's ID.

At step 820 the processor 500 of sponsor station 50 decrypts if necessary and applies $D_{USR}$ and $N_R$, which as discussed above have also been previously stored at the sponsor station 50, to decrypt and recover PR from sign (Sha-1(PL), $D_{USR}$), i.e. $\text{Decrypt}$(CSRP, $D_{USR}$), $E_R$, $N_R$, $\text{sign}$(Sha-1(PL), $D_{USR}$). In step 825, the processor 500 verifies that the challenge included in this recovered PR is correct and, if so, thereby explicitly verifies the user's knowledge of the factor $D_{USR}$. If not, the processor 500 suspends the user's ID. The successful completion of steps 815 and 825 explicitly verifies the user's knowledge of the user's key portion $D_{L1}$ and the factor $D_{USR}$.

Furthermore, the successful completion of step 810-825 implicitly verifies the user's knowledge of the user's password, and hence also of factor $F_p$ and other factors, if any, required to correctly construct $D_L$. It will be recognized that, if desired, the user's knowledge of any of such other factors could be verified implicitly by the sponsor station 50 in a manner similar to that described with reference to steps 820 and 825, although this is not mandatory, and one or more of such direct additional verifications may be omitted depending on the implementation.

In this implementation, upon successful completion of step 825, the user is authenticated and the login is completed. Following completion of the login, the sponsor station 50 and the user device 30 interact to create a symmetric session crypto-key used to encrypt and MAC communications between the two. Creation and use of such a session key is well known in the art and according to will not be further described here.

As described above, the use of rolling key to secure one or more factor needed to generate a private key portion of a user's split asymmetric crypto-key decreases the ability of an attacker to spoof the system. Even in the unlikely event that an attacker steals or otherwise obtains access to the encrypted factor, e.g. $CD_{USR}$, the rolling key, e.g. $D_R$ and the user's ID and password, the rolling key $D_R$ will become outdated the next time the user performs a login, whether it is successful, which will be the case if the user's login is performed prior to the attacker attempting to use the information to login as the user, or unsuccessful, which will be the case if the user's login is performed after the attacker has used the information to login as the user. This is because, in either case, the rolling key...
obtained by the attacker and that maintained by the user will
be out of synchronization after it is used one time for login.

Furthermore, if the user’s login is performed prior to the
attacker attempting to login using the information, the rolling
key that the attacker has access to will be out of synchroni-
ization with the corresponding key maintained by the authen-
ticating network, device, e.g. the sponsor station 50. On the
other hand, if the user’s login is performed after the attacker
has logged-in using the information, the rolling key that the
user has access to will be out of synchronization with the
corresponding key maintained by the authenticating network
device, e.g. the sponsor station 50. Thus the next attempted
login, in the former case by the attacker and in the later case
by the user, will result in an immediate suspension of the
user’s ID and thus the user’s account.

In summary, the rolling key protection is difficult for an
attacker to defeat. Furthermore, the use of an asymmetric
crypto-key as the rolling key, with one of the asymmetric key
pair stored by the entity other than the user, makes the system
even more difficult for an attacker to overcome.

The encryption of a factor, such as $D_{ESUP}$, which serves as a
first line of protection, in and of itself enhances the security,
since the factor is never persistently stored in the clear. Even
if a rolling key, such as $D_R$, is stored in the clear, which is not
preferred, another rolling key, such as $E_R$, is still required to
actually make use of $D_{ESUP}$, and therefore provides a second
line of protection. The preferred encryption of the rolling key,
such as $D_R$, will serve as a third line of protection and further
enhance the security, since in such case the rolling key is never
persistently stored in the clear. Yet, if all else fails, there is
also another line of protection. More particularly, even if,
for example, an attacker were to successfully steal the remov-
able media from user and attempt to use it to successfully
login, the attack would be foiled if the attacker does not have
the user’s ID, or the user’s password or any other factors
required to generate the applicable key portion of a split
private, or for that matter public, key.

However, even if the attacker were to overcome all of the
above obstacles, the robbing of the rolling keys make it
highly likely that the attacker would, at best, be able to spoof
the system only one time before the rolling key which the
attacker has access to becomes useless or the legitimate user
becomes aware of the attack because the rolling keys are no
longer in sync. Thus, an attack using the rolling key stored by
a user on a removable media or elsewhere, should quickly
become apparent to the legitimate user and/or authenticating
entity and result in prompt suspension of the user’s ID and
account after an attack, at least until the rolling key is reset.

It will also be recognized by those skilled in the art that,
while the invention has been described above in terms of one
or more preferred embodiments, it is not limited thereto.
Various features and aspects of the above described invention
may be used individually or jointly. Further, although the
invention has been described in the context of its implementa-
tion in a particular environment and for particular purposes,
e.g. in providing security for network communications, and,
more particularly, for securing factors of split private asym-
metric keys, those skilled in the art will recognize that its
usefulness is not limited thereto and that the present invention
can be beneficially utilized in any number of environments
and implementations. Accordingly, the claims set forth below
should be construed in view of the full breath and spirit of the
invention as disclosed herein.

What is claimed is:
1. A system for securing information on a communications
network comprising:
a storage device configured to store information encrypted
with a first type-one rolling key, wherein the first type-
one rolling key and a first type-two rolling key are associ-
ated with a first asymmetric rolling crypto-key; and
a processor configured with logic to (i) direct transmission
from a sending entity to an external computer system,
via a network, of a request for the first type-two rolling
key; (ii) receive from the external computer system, via
the network, the first type-two rolling key responsive to
the directed transmission, wherein the external com-
puter system sends the first type-two rolling key after
authentication of the sending entity based on a transmis-
sion of a proof of knowledge of the first type-one rolling
key from the sending entity to the external computer
(ii) decrypt the stored encrypted information with the first
type-two rolling key, (iv) after so decrypting the informa-
tion, generate a second asymmetric rolling crypto-
key having a second type-one rolling key and a second
type-two rolling key, (v) encrypt the information with
the second type-one rolling key

wherein at least one of the first type-one rolling key and the
second type-one rolling key and the second type-one rolling key are generated based on a
factor, which is generated based at least on the information
or a password associated with a user.

2. The system according to claim 1, wherein:
the processor is further configured with logic to (i) generate
the first asymmetric rolling crypto-key, and (ii) encrypt
the information with the type-one first rolling key; and
the storage device is configured to store the information
encrypted with the first type-one rolling key or second
type-one rolling key by the processor.

3. The system according to claim 1, wherein the processor
is further configured with logic to generate the first type-one
rolling key and the second type-one rolling key based on the
factor, and at least one of the first asymmetric rolling crypto-
key and the second asymmetric rolling crypto-key are
associated with at least one of a user and the sending entity.

4. The system according to claim 1, wherein:
the processor is further configured with logic to (i) direct
transmission to an external computer system, via the
network, of a request for the second type-two rolling
key, (ii) receive from the external computer system, via
the network, the second type-two rolling key responsive to
the directed transmission, (iii) decrypt the stored information encrypted with the second type-one rolling
key by application of the received second type-two roll-
ing key, (iv) after so decrypting the information, gener-
ate a third asymmetric rolling crypto-key having a third
type-one rolling key and a third type-two rolling key, (v)
encrypt the information with one of the third type-one
rolling key and the third public rolling key, and (vi)
transmission to an external computer system, via
the network, of a request for the third type-two rolling
key; and

the storage device is further configured to store the informa-
tion encrypted with the third type-one rolling key.

5. The system according to claim 1, wherein the storage
device is a portable storage device configured to be intercon-
ected with and to be disconnected from the processor.

6. A system for securing information on a communications
network, comprising:
a storage device configured to store information encrypted
with a first type-one rolling key, wherein the first type-

one rolling key and a first type-two rolling key are associated with a first asymmetric rolling crypto-key; and
a processor configured with logic to (i) direct transmission from a sending entity to an external computer system,
via a network, of a request for the first type-two rolling key, (ii) receive from the external computer system, via
the network, the first type-two rolling key responsive to the directed transmission, wherein the external com-
puter system sends the first type-two rolling key after authentication of the sending entity based on a transmis-
sion of a proof of knowledge of the first type-one rolling key from the sending entity to the external computer (iii)
decrypt the stored encrypted information with the first type-two rolling key, (iv) after so decrypting the informa-
tion, generate a second asymmetric rolling crypto-key having a second type-one rolling key and a second type-two rolling key, and (v) encrypt the information with the second type-one rolling key;
wherein the storage device is further configured to store the information encrypted with the second type-one rolling key;
wherein at least one of the first type-one rolling key and the second type-one rolling key is generated based on a factor, which is generated based at least on the information or a password associated with a user;
wherein at least one of the first type-one rolling key and the second type-one rolling key is a split private key having a first private key portion and a second private key portion; and
wherein the processor is further configured with logic to generate the first private key portion of at least one of the
first type-one rolling key and the second type-one rolling key associated with the user based on the factor.
7. A method for securing information on a communications network, comprising:
storage information encrypted with a first type-one rolling key, wherein the first type-one rolling key and a first
type-two rolling key are associated with a first asymmetric rolling crypto-key, wherein the stored information is accessible to a first network entity;
transmitting, via a network by the first network entity to a second network entity, a request for the first type-two rolling key;
transmitting, via the network by the first network entity to the second network entity, proof of knowledge of the first type-one rolling key, to authenticate the first network entity to the second network entity prior to receipt of the first type-two rolling key;
receiving, via the network by the first network entity from the second network entity, the requested first type-two rolling key;
decrypting, by the first network entity, the stored information encrypted with the first type-one rolling key, by application of the received first type-two rolling key;
generating, by the first network entity, a second asymmetric rolling crypto-key having a second type-one rolling key and a second type-two rolling key;
encrypting, by the first network entity, the information with the second type-one rolling key; and
storing the information encrypted with the second type-one rolling key,
wherein at least one of the first type-one rolling key and the second type-one rolling key are generated based on a factor, which is generated based at least on the information or a password associated with a user.
8. The method according to claim 7, wherein at least one of the first type-one rolling key and the second type-one rolling key is a private key and at least one of the first type-two rolling key and the second type-two rolling key is a public key.
9. The method according to claim 7, further comprising:
transmitting, via the network from the first network entity to the second network entity, the requested type-
two rolling key;
receiving, via the network by the first network entity from the second network entity, the requested type-
two rolling key;
decrypting, by the first network entity, the stored information encrypted with the second type-one rolling key, by application of the received type-one rolling key;
generating, by the first network entity, a third asymmetric rolling crypto-key having a third type-one rolling key and a third type-two rolling key;
encrypting, by the first network entity, the stored information with the third type-one rolling key; and
storing the information encrypted with the third type-one rolling key.
11. A method for securing information on a communications network, comprising:
storage information encrypted with one of a first type-one rolling key of a first asymmetric rolling crypto-key,
wherein the first type-one rolling key and a first type-two rolling key are associated with the first asymmetric rolling crypto-key associated with a first network entity, wherein the stored information is accessible to the first network entity, wherein at least one of the first type-one rolling key and the first type-two rolling key is generated based on a factor, which is generated based at least on the information or a password associated with a user and the at least one of the first type-one rolling key and the first type-two rolling key is a split private key having a first private key portion and a second private key portion;
generating the factor based on at least the information or a password associated with a user, and encrypting the factor with the first type-one rolling key;
transmitting, via a network by the first network entity to a second network entity, a request for the first type-two rolling key;
receiving, via the network by the first network entity from the second network entity, the requested type-one rolling key;
decrypting, by the first network entity, the factor encrypted with the first type-one rolling key, by application of the received first type-two rolling key;
generating, by the first network entity, a second asymmetric rolling crypto-key having a second type-one rolling key and a second type-two rolling key;
encrypting, by the first network entity, the factor with the second type-one rolling key;
encrypting, by the first network entity, the first private key portion based on the factor and another factor cor-
responding to a password associated with the first network entity; and
transmitting, via the network from the first network entity to the second network entity, proof of knowledge of the first private key portion to authenticate the first network entity to the second network entity.
12. An article of manufacture for securing information on a communications network, comprising:
computer readable storage media; and
logic stored on the storage media, wherein the stored logic
is configured to be readable by a computer and thereby
cause the computer to operate so as to:
store information encrypted with one of a first type-one
rolling key, wherein the first type-one rolling key and a
first type-two rolling key are associated with a first
asymmetric rolling crypto-key, and the stored informa-
tion is accessible to a first network entity;
transmit, via a network to a second network entity, a
request for the first type-two rolling key;
transmit, via the network to the second network entity,
proof of knowledge of the first type-one rolling key, to
authenticate the first network entity to the second net-
work entity, wherein the second network entity sends the
first type-two rolling key to the first network entity only
after the authentication is successful;
receive, via the network from the second network entity,
the requested first type-two rolling key;
decrypt the stored information encrypted with the first
type-one rolling key, by application of the received the
first type-two rolling key;
generate a second asymmetric rolling crypto-key having a
second type-one rolling key and a second type-two roll-
ing key;
encrypt the information with the second type-one rolling
key; and
store the information encrypted with the second type-one
rolling key,
wherein at least one of the first type-one rolling key and the
second type-one rolling key are generated based on a
factor, which is generated based at least on the informa-
tion or a password associated with a user.

13. The article of manufacture according to claim 12,
wherein the stored logic is further configured to cause the
computer to operate so as to (i) generate the first asymmet-
ric rolling key having the first type-one rolling key and the first
type-two rolling key, and (ii) encrypt the information with the
first type-one rolling key.

14. The article of manufacture according to claim 12,
wherein the stored logic is further configured to cause the
computer to operate so as to:
transmit, via the network to the second network entity, the
second type-one rolling key;
transmit, via the network to the second network entity, a
request for the second type-two rolling key after having
transmitted the second type-one rolling key;
receive, via the network from the second network entity,
the requested the second type-two rolling key;
decrypt the stored information encrypted with the second
type-one rolling key, by application of the received the
second type-two rolling key;
generate a third asymmetric rolling crypto-key having a
third type-one rolling key and a third type-two rolling
key;
encrypt the information with the third type-one rolling key;
and
store the information encrypted with the third type-one
rolling key.

15. An article of manufacture for securing information on a communications network, comprising:
computer readable storage media; and
logic stored on the storage media, wherein the stored logic
is configured to be readable by a computer and thereby
cause the computer to operate so as to:
store information encrypted with a first type-one rolling
key, wherein the first type-one rolling key and a first
type-two rolling key are associated with a first asymmet-
ric rolling crypto-key, and the stored information is
accessible to a first network entity;
transmit, via a network to a second network entity, a
request for the first type-two rolling key;
receive, via the network from the second network entity,
the requested first type-two rolling key;
decrypt the stored information encrypted with the first
type-one rolling key, by application of the received the
first type-two rolling key;
generate a second asymmetric rolling crypto-key having a
second type-one rolling key and a second type-two roll-
ing key;
encrypt the information with the second type-one rolling
key;
and
store the information encrypted with the second type-one
rolling key,
wherein at least one of the first type-one rolling key and the
second type-one rolling key are generated based on a
factor, which is generated based at least on the informa-
tion or a password associated with a user;
wherein at least one of the first type-one rolling key and the
first type-two rolling key is a type of a private key,
wherein the private key is a split private key having a first
private key portion and a second private key portion; and
wherein the stored logic is further configured to cause the
computer to operate so as to (i) generate the first private
key portion based at least on the factor and another factor
and (ii) transmit, via the network to the second network entity, proof of knowledge of the first
private key portion to authenticate the first network entity to the second network entity.

* * * * *