Password management: distribution, review and revocation

Lanfranco Lopriore

Dipartimento di Ingegneria dell’Informazione, Università di Pisa, via G. Caruso 16, 56126 Pisa, Italy
E-mail: l.lopriore@iet.unipi.it

Abstract — We consider the problem of access privilege management in a classical protection environment featuring subjects attempting to access the protected objects. We express an access privilege in terms of an access right and a privilege level. The privilege level and a protection diagram associated with each given object determine whether a nominal access privilege for this object corresponds to an effective, possibly weaker access privilege, or is revoked. We associate a password system with each object; the password system takes the form of a hierarchical bidimensional one-way chain. A subject possesses a nominal access privilege for a given object if it holds a key that matches one of the passwords in the password system of this object; the protection diagram determines the extent of the corresponding effective access privilege. The resulting protection environment has several interesting properties. A key reduction mechanism allows a subject that holds a key for a given object to distribute keys for weaker access rights at lower privilege levels. A subject that owns a given object can review or revoke the passwords for this object by simply modifying the protection diagram. The memory requirements to represent a protection diagram are negligible; as far as password storage is concerned, space-time trade-offs are possible.

Keywords: access privilege, one-way chain, one-way function, password, review, revocation.

1. INTRODUCTION

In a classical protection model, active entities, called subjects, attempt to access passive entities, the protected objects [20]. Objects are typed, and the type of a given object states the set of values that the object can assume, the set of the operations that can be applied to these values, and a set of access rights; for each operation, the definition of the object type states the access right that is necessary to execute that operation successfully.

1.1. Access privilege specification

In a system of this type, a basic problem is to specify the relations existing between subjects and protected objects by defining the access rights that each subject holds on the objects [21]. A classical solution is based on the concept of a capability [12]Levy. This is a pair (G, AR), where G is the name of an object and AR is an access privilege expressed in terms of a set of access rights. A subject that holds capability (G, AR) can access object G to perform the operations permitted by the access rights in AR.

Capabilities should be segregated in memory, so that their internal representation is inaccessible to subjects [3]. This is necessary to prevent a subject from tampering with an existing
capability to amplify the access rights it contains, or even forging a capability from scratch. Several solutions have been devised to the capability segregation problem. Special memory segments, which we shall call the capability segments, can be reserved for capability storage (in contrast, data segments will be reserved for storage of ordinary information items) [4], [9], [23]. This solution leads to segment proliferation. The resulting programming paradigm is often unnatural, e.g. the representation of a data structure must include a capability segment storing the capabilities for the data segments. Alternatively, capability segregation can be obtained by taking advantage of a tagged memory system that associates a 1-bit tag with each memory cell; the value of the tag of a given cell indicates whether this cell contains a capability or an ordinary information item [7], [18]. In this approach, the instruction set of the processor includes special instructions for capability processing; if one of these instructions is executed on a memory cell tagged to contain an ordinary information item, an exception of violated protection is raised and execution fails. Contrary to hardware standardization [16], ad-hoc memory modules are necessary to support cell tags (e.g. a 65-bit cell is used to store a 64-bit memory word).

In a different approach, a set of passwords is associated with each protected object, and each password corresponds to a subset of all access rights [1], [2], [6], [15], [19]. In order to access an object to perform a given operation, a subject presents a key for this object. The access is successful only if the key matches one of the passwords associated with the object, and this password includes the access right required by that operation; if this is not the case, an exception of violated protection is raised and the object access fails.

Keys are protected from forging by the password size; if passwords are large, the probability of guessing a valid key is vanishingly low. It follows that keys do not need to be segregated in memory; instead, they can be freely mixed with ordinary data. This simplifies software composition in memory. Furthermore, no need for special hardware is connected with the necessity to store cell tags.

1.2. Access right distribution

A subject S that holds a set of access rights for a given object G should be in a position to distribute these access rights, in full or in part, to other subjects. In a capability environment, a result of this type is obtained by a simple action of a capability copy (and indeed, simplicity in access right distribution was one of the original motivations for the introduction of the capability concept [12]).

In a password-based protection system, subject S that possesses a key $k$ for object G can
distribute the access privilege corresponding to this key to another subject by simply transmitting a copy of the key. A related problem is that of key reduction. Subject S should be given the ability to transmit only a subset of the access rights corresponding to key \( k \). This means that a mechanism of the protection system should allow S to convert \( k \) into a weaker key \( k' \). The key reduction problem can be easily solved by associating an object manager with each given object. In a situation of this type, subject S transmits key \( k \) to the object manager that verifies the validity of \( k \) and then generates a key \( k' \) matching a weaker password; \( k' \) is returned to S. Of course, the presence of an object manager tends to complicate the overall object structure, and is contrary to a main requirement of a protection system design, i.e. simplicity in access right management.

1.3. Access right review and revocation

A subject that granted an access privilege to another subject should be allowed to revise the grant and revoke the access privilege from the recipient. Revocation should extend to all the subjects that received the privilege being revoked from the first recipient, recursively. Revocation should be reversible, so that after a given access privilege has been revoked, it should be possible to confirm this access privilege and restore its validity.

Several solutions to the revocation problem have been proposed with reference to capability-based protection environments [14]. These include a reference monitor associated with the given object that manages the access permissions held by all subjects on this object [17], [22]; a propagation graph for each capability that keeps track of all successive transfers of this capability throughout the system [5]; and short-lived capabilities, whose validity must be renewed periodically, and are implicitly revoked if renewal is lacking [11]. These solutions tend to subvert the main advantage of the capability-based protection model, i.e. simplicity in access right transmission between subjects [13].

In a password-based protection environment, a subject that receives a key is free to transmit this key further, and this key diffusion process may well extend to any transition depth. This means that access rights tend to propagate throughout the system. A simple solution to the revocation problem is to modify one or more passwords associated with the given object, so that the corresponding keys lose their validity. This solution does not meet the requirement to limit revocation to a subset of the access rights associated with the given password.

This paper presents a model of a password management system that has been designed by taking the following requirements into account:

- A simple mechanism for access privilege distribution should permit effective forms of key
reduction, so that a subject that holds a given access privilege can transmit this privilege only in part.

- Password review and revocation should be fully supported, so that the extent of a given password can be restricted to a subset of the original access rights. The consequence of a password review should extend to all keys matching the reviewed password. It should be possible to revert the effect of a password review and restore full password validity.

- The memory requirements for storage of the passwords and the information items related to password management should be kept to a minimum.

The rest of this paper is organized as follows. Section 2 presents our protection model with special reference to access privileges and passwords. The concept of a protection diagram is introduced expressing a relation between the nominal access privilege connected with a password and the effective access privilege granted by possession of a key matching that password. Section 3 presents the password system with special reference to password generation and management. Section 4 discusses the protection model from a number of salient viewpoints, which include the verification, review and revocation of access rights, and the memory requirements for storage of the passwords and the protection diagram. Section 5 gives concluding remarks.

2. THE PROTECTION MODEL

Let T be an object type, and let $op_0, op_1, \ldots$ be the operations that can be applied to the objects of type T. Furthermore, let $ar_0, ar_1, \ldots, ar_{r-1}$ be the access rights, and let $pl_0, pl_1, \ldots, pl_{c-1}$ be the privilege levels defined by T, where quantities $r$ and $c$ are specific to T. Privilege levels are involved in password validation, review and revocation, as will be illustrated shortly.

The access rights are ordered from the weakest $ar_0$ to the strongest $ar_{r-1}$. Possession of access right $ar_i$ implies possession of every weaker access right $ar_{i'}, i' < i$. Similarly, the privilege levels are ordered from the lowest $pl_0$ to the highest $pl_{c-1}$.

An access privilege for an object G of type T is the specification of an access right and a privilege level. Let $P_{\text{nom}} = (ar_i, pl_j), 0 \leq i \leq r - 1, 0 \leq j \leq c - 1$, be a nominal access privilege expressed in terms of access right $ar_i$ and privilege level $pl_j$. $P_{\text{nom}}$ grants its holder an effective access privilege $P_{\text{eff}} = (ar^*, pl_j)$ expressed in terms of access right $ar^*$ and the same privilege level $pl_j$, where $ar^*$ is not stronger than $ar_i$. The actual extent of $ar^*$ is stated for object G by a relation existing between the access rights and the privilege levels. This relation is specific to G, and is expressed by a diagram, which is associated with G and is called the protection diagram of G.
The protection diagram is defined in a two-dimensional Cartesian coordinate system. In the protection diagram, the horizontal axis refers to the access rights, from $ar_0$ to $ar_{r-1}$, and the vertical axis refers to the privilege levels, from $pl_0$ to $pl_{c-1}$. The protection line is a polygonal chain whose vertices are access privileges that we call the limit privileges, with the constraint that at least one limit privilege must be associated with each access right. Figure 1 shows examples of protection diagrams featuring four access rights, $ar_0$ to $ar_3$, and five privilege levels, $pl_0$ to $pl_4$. The protection line divides the protection diagram into two regions: a validity region that includes the protection line and corresponds to higher privilege levels, and a downgrade region that corresponds to lower privilege levels.

In the example of Figure 1a, the limit privileges are $(ar_0, pl_1)$, $(ar_1, pl_2)$, $(ar_2, pl_2)$, and $(ar_3, pl_4)$. The protection line connects the corresponding points of the protection diagram. The validity region occupies the upper-left part of the diagram and includes the protection line; the downgrade region occupies the lower-right part. In Figure 1b, the limit privileges are $(ar_0, pl_4)$, $(ar_1, pl_3)$, $(ar_1, pl_2)$, $(ar_2, pl_2)$, and $(ar_3, pl_1)$. The validity region occupies the upper-right part...
of the diagram and includes the protection line; the downgrade region occupies the lower-left part. In Figure 1c, the protection line is horizontal and connects limit privileges \((ar_0, pl_4)\), \((ar_1, pl_4)\), \((ar_2, pl_4)\), and \((ar_3, pl_4)\). The validity region degenerates into the protection line. Finally, in Figure 1d the protection line is horizontal and connects limit privileges \((ar_0, pl_0)\), \((ar_1, pl_0)\), \((ar_2, pl_0)\), and \((ar_3, pl_0)\). In this case, the validity region occupies the whole protection diagram.

For the given nominal access privilege, the protection line and the two protection regions determine the corresponding effective privilege. Let us refer again to nominal privilege \(P_{nom} = (ar_i, pl_j)\) and the corresponding effective privilege \(P_{eff} = (ar^*, pl_j)\). If \(P_{nom}\) is in the validity region (including the protection line), nominal access right \(ar_i\) is effective, that is, \(ar^* = ar_i\) and consequently \(P_{eff} = P_{nom}\); a subject \(S\) that possesses \(P_{nom}\) is granted access right \(ar_i\). If \(P_{nom}\) is in the downgrade region, access right \(ar_i\) is downgraded to the strongest access right \(ar^*\) at protection level \(pl_j\) that is weaker than \(ar_i\) and is included in the validity region. If \(ar_i\) cannot be downgraded into the validity region, then access privilege \(P_{nom}\) grants no access right at all; in a situation of this type, we say that \(P_{nom}\) is revoked.

In the example of Figure 1a, nominal access privilege \((ar_1, pl_3)\) is in the validity region, and is effective; a subject that possesses this access privilege is granted access right \(ar_1\). Nominal privilege \((ar_3, pl_3)\) is in the downgrade region, and the corresponding effective privilege is \((ar_2, pl_3)\); a subject that possesses \((ar_3, pl_3)\) is granted access right \(ar_2\). Similarly, nominal privilege \((ar_2, pl_1)\) is in the downgrade region, and the corresponding effective privilege is \((ar_0, pl_1)\). Finally, nominal privilege \((ar_1, pl_0)\) is in the downgrade region and cannot be downgraded into the validity region, so it is revoked (this access privilege grants no access right at all). In the example of Figure 1b, nominal privileges \((ar_3, pl_3)\) and \((ar_3, pl_1)\) are in the validity region, and are effective. Nominal privilege \((ar_1, pl_1)\) cannot be downgraded into the validity region, and is revoked. In the example of Figure 1c, all nominal privileges are revoked except those defined in terms of the highest privilege level, \(pl_4\), which are effective. Finally, in the example of Figure 1d, the validity region occupies the entire protection diagram, so all nominal privileges are effective.

In every given object type, the strongest access right \(ar^{-1}\) is called the own access right; possession of this access right for a given object makes it possible to delete the object and to modify its protection line (that is, to modify the position of the limit privileges in the protection diagram). Access privilege \((ar^{-1}, pl_{c^{-1}})\) corresponding to the own access right at the highest privilege level is called the owner privilege. A subject that creates a new object (the object owner) is granted the owner privilege for this object. When an object is created, a protection diagram is associated with that object, and the shape of the protection line is determined as part
of the object initialization.

An important property is that, in the protection diagram, all privileges at privilege level \( pl_{c-1} \) are always in the validity region; it follows that the owner privilege is always effective. Consequently, a subject that creates an object is always in a position to delete this object and to modify its protection line; as will be shown shortly, an action of this type corresponds to a form of review or revocation of access privileges.

3. THE PASSWORD SYSTEM

Let us refer again to type \( T \), where \( ar_0, ar_1, \ldots, ar_{r-1} \) are the access rights, \( pl_0, pl_1, \ldots, pl_{c-1} \) are the privilege levels, and \( P_{\text{nom}} = (ar_i, pl_j), 0 \leq i \leq r - 1, 0 \leq j \leq c - 1 \), are the nominal access privileges. We shall refer to the generic operation \( op \) defined by \( T \), \( ar_{op} \) being the access right necessary to execute this operation successfully. In our protection model, for each object \( G \) of type \( T \), a password system is associated with \( G \) featuring a password for each nominal access privilege, for a total of \( r \cdot c \) passwords. In order to access \( G \) to execute operation \( op \) successfully, a given subject \( S \) must present a key \( k \) matching a password in the password system, say password \( w_{ij} \). Implementation of operation \( op \) includes the actions necessary to determine the effective access privilege \( P_{\text{eff}} = (ar^*, pl_j) \) corresponding to the nominal access privilege \( P_{\text{nom}} = (ar_i, pl_j) \) associated with \( w_{ij} \); to this aim, \( op \) uses the protection diagram of \( G \). Execution of operation \( op \) will terminate successfully only if \( ar^* \) is stronger than or equal to access right \( ar_{op} \); if this is not the case, execution terminates with failure.

A related problem is the generation of the set of passwords for object \( G \). We solve this problem by taking advantage of one-way chains.

3.1. Password generation

Function \( F \) is one-way if it is easy to compute but hard to invert [10]. This means that given a value \( x \) it is computationally easy to compute \( F(x) \), but given a value \( y \) it is computationally unfeasible to find a value \( x \) such that \( y = F(x) \). We shall denote \( i \) successive applications of \( F \) by \( F^i \), e.g. \( F^2(x) = F(F(x)) \). A one-way chain [8] is a collection of values \( (v_0, v_1, \ldots, v_{n-1}) \) such that each \( v_i \) except the last value \( v_{n-1} \) is the result of applying a one-way function \( F \) to the next value, i.e. \( v_i = F(v_{i+1}) = F^{n-i-1}(v_{n-1}) \) for \( 0 \leq i < n - 1 \). Value \( v_{n-1} \) is called the seed of the chain. A hierarchical bidimensional one-way chain consists of two levels of chains, where each value of the chain at the first level (the primary chain) is the seed of a chain at the second level (a secondary chain).

In our model, the password system of object \( G \) of type \( T \) is configured as a hierarchical
bidimensional one-way chain (Figure 2). To set up the password system, password $w_{r-1,c-1}$, corresponding to owner privilege $(ar_{r-1}, plc_{r-1})$, is chosen at random and is the seed of the primary chain. A one-way function $PF$, called the primary function, is used to generate the successive passwords in the primary chain, which are associated with access right $ar_{r-1}$ in the direction of decreasing privilege levels, i.e. $w_{r-1,c-2} = PF(w_{r-1,c-1}), w_{r-1,c-3} = PF(w_{r-1,c-2}) = PF^2(w_{r-1,c-1}), \ldots, w_{r-1,0} = PF(w_{r-1,1}) = PF^{r-1}(w_{r-1,c-1})$.

Each given password $w_{r-1,j}$ of the primary chain is the seed of the secondary chain associated with the corresponding privilege level, $pl_j$. The secondary chain features $r$ values, one value for each access right. A further one-way function, the secondary function $SF$, is used to setup the secondary chains, so that for privilege level $pl_j$ we have $w_{r-2,j} = SF(w_{r-1,j}), w_{r-3,j} = SF(w_{r-2,j}) = SF^2(w_{r-1,j}), \ldots, w_{0,j} = SF(w_{1,j}) = SF^{r-1}(w_{r-1,j})$.

### 3.2. Password distribution

As seen in Section 1, the problem of access privilege distribution consists in allowing a subject that holds an access privilege for a given object to grant this access privilege to a different subject. A salient feature of password-based protection systems is simplicity in access privilege distribution. In a system of this type, a subject $S$ that holds a key $k$ for a given object is in a position to distribute the corresponding access privilege to another subject $S'$, by simply transmitting a copy of $k$ to $S'$.

In our protection system, subject $S$ may even distribute an access privilege weaker than the privilege in the original key. Key reduction is the action of transforming a key for a given access privilege into a key for a reduced privilege. Key reduction can take place in the direction of a weaker access right, a lower privilege level, or both. Key reduction is obtained by taking advantage of the structure of the password system and the password chains. In detail:

![Figure 2. The password system featuring a primary chain whose seed is the password $w_{r-1,c-1}$ of the owner privilege, and a secondary chain for each privilege level, whose seed is the corresponding element of the primary chain.](image-url)
A subject that holds key $k$ matching password $w_{r-1,j}$ for access privilege $(ar_{r-1}, pl_j)$ expressed in terms of the strongest access right $ar_{r-1}$ and privilege level $pl_j$, can derive a key $k'$ matching password $w_{r-1,j'}$ for access privilege $(ar_{r-1}, pl_{j'})$ expressed in terms of the same access right $ar_{r-1}$ and a lower privilege level $pl_{j'}, j' < j$. To this aim, the subject will apply primary one-way function $PF$ iteratively $j - j'$ times, i.e. $k' = PF^{j'-j}(k)$. This is a consequence of the fact that $w_{r-1,j'} = PF^{j'-j}(w_{r-1,j})$.

A subject that holds key $k$ matching password $w_{i,j}$ for access privilege $(ar_i, pl_j)$ expressed in terms of access right $ar_i$ and privilege level $pl_j$ can derive a key $k'$ matching password $w_{i,j'}$ for access privilege $(ar_{i'}, pl_{j})$ expressed in terms of a weaker access right $ar_{i'}, i' < i$, and the same privilege level $pl_j$. To this aim, the subject will apply secondary one-way function $SF$ iteratively $i - i'$ times, i.e. $k' = SF^{i'-i}(k)$. This is a consequence of the fact that $w_{i,j'} = SF^{i'-i}(w_{i,j})$.

### 3.3. Password review and revocation

As a consequence of the inherent simplicity of key distribution, in a password-based protection environment access privileges tend to spread throughout the system. A related problem is to allow the owner of a given object to control the extent of the effective access privilege granted by each password on that object, in contrast with the nominal access privilege corresponding to that password (password review). The object owner should be even in a position to revoke the nominal access privilege, so that the password can no longer be used for successful object access (password revocation).

In our design, the object owner can change the position of the limit privileges in the protection diagram, thereby modifying the protection line and the configuration of the validity region and the downgrade region. So doing, the object owner can enforce specific access privilege management strategies, by reviewing or revoking each given password, independently of the subjects that hold this password. In the following, we shall say that a password is effective, downgraded or revoked if this is the case for the corresponding nominal access privilege.

For instance, in the protection diagram of Figure 1d, the protection line is horizontal and is placed at privilege level 0. Consequently, all nominal access privileges, and the corresponding passwords, are effective. In a situation of this type, a subject that holds a key matching a password corresponding to a given access privilege can exercise the full access right included in this access privilege. By moving the protection line to a high privilege level, e.g. $pl_4$ (see Figure 1c), the object owner revokes all the passwords corresponding to nominal access privileges at lower privilege levels. A subject that holds a key matching one of these passwords cannot use
this key for successful object access. If the protection line is that of Figure 1a, high privilege levels are necessary to exercise strong access rights. For instance, access privilege \((ar_3, pl_3)\) is below the protection line. This access privilege is downgraded to \((ar_2, pl_3)\); a subject that holds a key matching password \(w_{3,3}\) can use this key to execute a given operation only if this operation requires access right \(ar_2\) or lower. Finally, if the protection line is configured as shown in Figure 1b, the access privileges defined in terms of strong access rights are effective even at low privilege levels, e.g. \((ar_3, pl_1)\) and the corresponding password \(w_{3,1}\).

4. DISCUSSION

4.1. Access right verification

Let us refer again to object G of type T, let \(op\) be an operation defined by T, and let \(ar_{op}\) be the access right required for successful execution of \(op\). The password system of G is stored as part of the object internal representation. Now suppose that subject S tries to execute operation \(op\) on G. To this aim, S presents a key \(k\) to \(op\). The actions involved in the execution of \(op\) include key verification, which will be conceptually structured as follows:

1. Key \(k\) is compared with the passwords in the password system of object G. If no matching password is found, execution of operation \(op\) terminates with failure; otherwise, let \(w_{i,j}\) be the matching password, and let \(P_{\text{nom}} = (ar_i, pl_j)\) be the corresponding nominal access privilege, which is expressed in terms of access right \(ar_i\) and privilege level \(pl_j\).

2. The protection diagram of object G is used to determine whether \(P_{\text{nom}}\) is revoked. If this is the case, execution of operation \(op\) terminates with failure; otherwise, let \(P_{\text{eff}} = (ar^*, pl_j)\) be the effective access privilege corresponding to \(P_{\text{nom}}\).

3. Access right \(ar^*\) is compared with access right \(ar_{op}\) (it should be recalled that access rights are ordered hierarchically, so that every given access right includes all weaker access rights). If \(ar^* < ar_{op}\), execution of \(op\) terminates with failure; otherwise, access right verification is successful and the actions involved in the execution of \(op\) are subsequently accomplished.

4.2. Password review and revocation

As seen in Section 3.3, in our protection system a subject that owns a given object can review or revoke every password defined for that object by simply modifying the shape of the protection line. This mechanism results to possess a number of interesting properties:

- Password review can be limited to a subset of all passwords. This is an inherent property
of the protection diagram. For instance, in the protection diagram of Figure 1a, access privilege \((ar_3, pl_3)\) is downgraded to \((ar_2, pl_3)\); this means that password \(w_{3,3}\) grants access right \(ar_2\) instead of the nominal \(ar_3\). In contrast, access privilege \((ar_1, pl_3)\) is effective, and the extent of password \(w_{1,3}\) is unaltered; this password grants access right \(ar_1\).

- Similarly, password revocation can be limited to a subset of all passwords. In the example of Figure 1b, all the access privileges in the downgrade region are revoked, and consequently, the corresponding passwords cannot be used for successful object access, e.g. password \(w_{1,1}\) corresponding to access privilege \((ar_1, pl_1)\). On the other hand, the passwords for the access privileges in the validity region maintain their access right strength, e.g. password \(w_{3,3}\) corresponding to access privilege \((ar_3, pl_3)\).

- Two or more passwords corresponding to the same access right at different privilege levels can be revoked independently of each other. In the example of Figure 1a, password \(w_{1,3}\) corresponding to access privilege \((ar_1, pl_3)\) for access right \(ar_1\) at privilege level \(pl_3\) is in the validity region and is effective, whereas password \(w_{1,0}\) corresponding to access privilege \((ar_1, pl_0)\) for the same access right \(ar_1\) at privilege level \(pl_0\) is in the downgrade region and is revoked.

- The effects of a review or revocation are transitive [5], that is, they propagate to all the keys matching a given password independently of the subjects that hold these keys and the propagation paths followed by the keys to reach these subjects. If a given password is revoked or downgraded to a given extent, all keys matching this password are automatically revoked or downgraded to the same extent. In fact, a copy of a given key is indistinguishable from the original, and keys have no memory of the consecutive copy actions.

- The effects of a review or revocation are temporal [5], that is, they can be reversed through the same mechanism used for the revocation. With reference to Figure 1c, password \(w_{2,2}\) corresponds to a nominal access privilege in the downgrade region, and is revoked. If the protection line is moved to privilege level \(pl_2\), for instance, the nominal access privilege will be in the validity region, and the validity of \(w_{2,2}\) will be restored to its full extent.

4.3. Memory requirements

The protection line

As seen in Section 2, the shape of the protection line is completely determined by the coordinates of the limit privileges; at least one, and at most two limit privileges correspond to each access right (two limit privileges are necessary to delimit a vertical protection line segment). In a possible representation for up to 16 privilege levels, we shall reserve one byte for
each access right. The two 4-bit nibbles of the $i$-th byte encode the privilege levels of the two limit privileges for the $i$-th access right; if only one limit privilege corresponds to a given access right, the two nibbles will specify the same privilege level.

Let us refer to a protection diagram featuring four access rights, $aro$ to $ar_3$, and five privilege levels, $plo$ to $pl_4$; in this case, the shape of the protection line can be encoded in 4 bytes. Examples are given in Figures 1a to 1d; in these examples, the corresponding hexadecimal configurations are $11\ 22\ 22\ 44\$, $44\ 32\ 22\ 11\$, $44\ 44\ 44\ 44\$, and $00\ 00\ 00\ 00\$, respectively. We may conclude that the memory requirements to represent a protection line are negligible.

**Passwords**

The passwords in the password system of a given object will be evaluated when this object is initialized. These passwords will be stored in a password array that is part of the internal representation of the object. As seen in Section 4.1, the password system is used when an action of access right verification takes place, to find the password $w_{ij}$ matching a given key $k$. To this aim, a sequential search in the password array will check the array elements until the matching password is found. The expected number of comparisons is $(r \cdot c +1) / 2$, $r$ and $c$ being the number of access rights and the number of privilege levels, respectively.

In an alternative approach, a key has the form of a triple $(k, i, j)$, that is, key value $k$ and the indexes $i$ and $j$ of the matching password $w_{ij}$. In this case, a single comparison is necessary in the password array, at element $(r \cdot i + j)$ (here, we hypothesize that passwords are stored in the password array by privilege levels); if the value of this element does not match quantity $k$, key verification fails. The memory requirements for storage of quantities $i$ and $j$ are negligible, e.g. a single byte for up to 16 access rights and 16 privilege levels.

If memory is an extremely scarce resource, we shall store a single password instead of the entire password system, i.e. password $wr_{-1,c-1}$ corresponding to the seed of the primary chain (see Section 3.1 and Figure 2). In this case, verification of the validity of a given key requires recalculation of several passwords, which will be accomplished by using primary one-way function $PF$ and secondary one-way function $SF$. Specifically, we shall use $PF$ to evaluate passwords $wr_{-1,c-2}, wr_{-1,c-3}, \ldots,$ to find the seed $wr_{-1,j}$ of the $j$-th secondary chain; then, we shall use $SF$ to evaluate passwords $wr_{-2,j}, wr_{-3,j}, \ldots,$ to find the value of password $w_{ij}$. The expected number of password evaluations is $(c + r) / 2$.

A space-time trade-off is possible, by using the password array to permanently store only the passwords $w_{r-1,c-1}, w_{r-1,c-2}, \ldots, w_{r-1,0}$ that form the primary chain of the password system. In this case, verification of the validity of a key matching password $w_{ij}$ will require recalculation
of the first $r - i - 1$ passwords in the $j$-th secondary chain, whose seed is password $w_{r-1,j}$ in the primary chain. In this case, the expected number of password evaluations decreases to $r / 2$.

5. CONCLUDING REMARKS

We have considered the problem of access privilege management in a classical protection environment featuring subjects attempting to access the protected objects. We express an access privilege in terms of an access right and a privilege level. The privilege level and a protection diagram associated with each given object determine whether a nominal access privilege for this object corresponds to an effective, possibly weaker access privilege, or is revoked.

We associate a password system with each object. The password system takes the form of a hierarchical bidimensional one-way chain: a primary chain contains the passwords for the strongest access right at the different privilege levels; a secondary chain for each privilege level contains the passwords for the access rights at that privilege level. A subject possesses a nominal access privilege for a given object if it holds a key matching one of the passwords in the password system of that object; the protection diagram determines the extent of the corresponding effective access privilege.

The following is a brief summary of the main results we have obtained:

- A subject that holds a key for a given object is free to distribute this key to other subjects, which acquire the corresponding access privilege. A key reduction mechanism allows a subject that holds a key for the strongest access right at a given privilege level to distribute keys for the same access right or weaker access rights, at the same privilege level or at lower privilege levels. Furthermore, a subject that holds a key for an access right which is not the strongest access right can distribute keys for weaker access rights at the same privilege level.

- A subject that owns a given object can review or even revoke the passwords for this object by simply modifying the protection diagram. Review and revocation can be limited to a subset of all the passwords. Two or more passwords for the same access right at different privilege levels can be revoked independently of each other. We have shown that the effects of a password review or revocation are transitive and temporal.

- The memory requirements to represent a protection diagram are negligible. As far as password storage is concerned, space-time trade-offs are possible. If we store all the passwords of the password system as part of the internal representation of the given object, a single comparison is necessary to ascertain the validity of a given key. Alternatively, we can store
only the passwords in the primary chain, or even a single password, i.e. the seed of the primary chain, at the cost of recalculating several passwords to validate a key.

ACKNOWLEDGMENT

This work has been partially supported by the TENACE PRIN Project (Grant no. 20103P34XC_008) funded by the Italian Ministry of Education, University and Research.

REFERENCES


