Information Processing Letters 40 (1991) 235–239
North-Holland 13 December 1991

Distributed user identification by zero-knowledge access rights proving

Josep Domingo-Ferrer *

Departament d'Informàtica, Universitat Autònoma de Barcelona, E-08193 Bellaterra, Barcelona, Catalonia, Spain

Communicated by L. Boasson
Received 15 May 1991
Revised 21 October 1991

Abstract

Domingo-Ferrer, J., Distributed user identification by zero-knowledge access rights proving, Information Processing Letters 40 (1991) 235–239.

A scheme for identifying the rights of users to access files in a computer network with many servers is presented. Users are granted rights by an authority, and servers need only a certified list of available access rights in order to perform access control. A server stores no information about the users, neither access matrix nor passwords—*user portability*—, which enables the authority to perform user registration, rights granting and rights revocation independently of servers; moreover, the latter two are public operations in the clear. The number of rights shared by more than one user throughout the network is a lower bound for the number of secret pieces held by each user, but *his unshared rights can be increased indefinitely without changing or increasing his secret information*. Rights possession proofs are zero-knowledge and simultaneous.

Keywords: Distributed identification, access control, computer security, zero-knowledge proofs, distributed systems

1. Introduction

Consider a computer network in which there are a set of files, or more generally resources, a community of file users (i.e. workstation users) and a number of file servers whose task is to control user access to the files; users are granted or revoked access rights to files by a central authority or network security manager. If the user is an authorized one, but a requested file is not available to the server, the latter can either deny service or obtain the file from another server, depending on implementation.

We present a zero-knowledge based scheme that allows a user to convince any server in the network that he is authorized to access some of the network files. Our scheme has the following advantages against other systems (including public-key ones):

- In order to control user accesses, the file servers need only read access to a public certified list of available file access rights (which are records consisting of: file name, type of access and a public number y associated with this access). They do not need any information about the current users in the network (neither passwords nor access matrices). Thus the authority can register new users, give them rights, and revoke rights without communicating with each server.

* Now with the Centre de Supercomputació de Catalunya, Diagonal 645, E-08028 Barcelona, Catalonia, Spain.

- *Thanks to a linear transformation, the secret pieces held by a user are fixed.* The only constraint on their number is that it must be greater than the number of rights granted to more than one user throughout the network. This means that the authority can give a user as many unshared access rights as needed (in a file system most files are expected to be unshared) without having to change neither the user secret pieces nor their number.
- The *linear transformation* used allows also *granting rights to a user and revoking them in the clear*, without any confidential—i.e. encrypted—communication with the user (whose secret pieces remain unchanged!). Rights granting must respect the previous condition on the number of shared rights.
- When a user wants to show possession of rights to access a set of files, he can do it in a single zero-knowledge proof by proving knowledge of all his secret pieces (*which are constant and independent of the possessed rights*) simultaneously [1]; then the server temporarily records the identified rights (a sort of session capability list for the user), so that no further proof is needed during the rest of the session. Thus the scheme is fast and secure against eavesdropping.

2. A geometrical scheme

The mathematical structure used is similar to that of a linear geometrical threshold scheme [3]. Let A be the central network authority. Let p be a large prime and α a generator of $\mathcal{Z}/(p)^*$, both public and certified. Let t be an integer security parameter (see Subsection 2.1). Assume that A wants to give n rights to a user u, with $n, t \ll p$. Then A proceeds as follows:

Algorithm 1

1. Choose n random integers $x_0, \ldots, x_{n-1}$ over $\mathcal{Z}/(p-1)$.
2. Pick a random vector $V = (a_0, \ldots, a_{t-1})$ over $\mathcal{Z}/(p-1)$, such that all a_j's are nonzero and no two among them are equal. a_0 must be prime to $p-1$.
3. Find n vectors of size t, $Z_i = (z_{i0}, \ldots, z_{i(t-1)})$, with nonzero coordinates, for $0 \leqslant i \leqslant n-1$ over $\mathcal{Z}/(p-1)$, such that
$$\begin{aligned} x_0 &= a_0 z_{00} + \cdots + a_{t-1} z_{0(t-1)} \bmod (p-1) \\ &\vdots \\ x_{n-1} &= a_0 z_{(n-1)0} + \cdots \\ &\quad + a_{t-1} z_{(n-1)(t-1)} \bmod (p-1) \end{aligned} \tag{1}$$
To compute Z_i pick random nonzero z_{ij}, $j = 1, \ldots, t-1$ over $\mathcal{Z}/(p-1)$ and solve the ith equation for z_{i0} using that a_0 can be inverted over $\mathcal{Z}/(p-1)$. If $z_{i0} = 0$, then repeat its computation with a new choice for the other z_{ij}'s.
4. Compute $y_i := \alpha^{x_i} \bmod p$ for $i = 0$ to $n-1$. The meaning of each y_i is the right to access a particular file in a given way (no user information); knowledge of x_i is equivalent to possession of y_i. Publish the y_i's together with their meaning in a certified way.
5. Give the vectors Z_i, $0 \leqslant i \leqslant n-1$ to u in cleartext (no need for certification).
6. Give the vector V to u in a confidential way.

Now the following result holds:

Theorem 1. *If the authority A has completed Algorithm 1, then user u is able to show possession of his rights $y_0, \ldots, y_{n-1}$ (or a subset of them) to any server in the network, that need not previously know about him. The proof can be zero-knowledge and, no matter the value of n, it consists of simultaneously proving knowledge of t logarithms. Stealing a nongranted right is for an isolated user as hard as the discrete logarithm.*

Proof. User u supplies the server with integers z_{ij} $(\neq 0)$, A_j $(\neq 1)$, for $i = 0, \ldots, n-1$ and $j = 0, \ldots, t-1$, satisfying the following set of equations:
$$\begin{aligned} y_0 &= A_0^{z_{00}} A_1^{z_{01}} \cdots A_{t-1}^{z_{0(t-1)}} \bmod p \\ &\vdots \\ y_{n-1} &= A_0^{z_{(n-1)0}} A_1^{z_{(n-1)1}} \cdots A_{t-1}^{z_{(n-1)(t-1)}} \bmod p \end{aligned} \tag{2}$$

Now if u is able to prove his knowledge of $\log_\alpha A_j$ over $\mathcal{Z}/(p)$ for $j=0$ to $t-1$, then it follows that he is able to express the y_i's as powers of α, i.e. that he possesses $\log_\alpha y_i = x_i$ over $\mathcal{Z}/(p)$ for $i=0$ to $n-1$. But u has been given a_j, $j=0,\ldots,t-1$ in the last step of Algorithm 1, and it is straightforward from (1) that $A_j := \alpha^{a_j} \bmod p$, $j=0,\ldots,t-1$ satisfy (2) when the same z_{ij}'s are used in both systems. Now the protocoi [1] can be used to simultaneously show knowledge of the logarithms of A_j, $j=0,\ldots,t-1$ in zero knowledge.

As for security, (2) is verifiable by the server since the y_i's are public and certified. On the other hand, if u does not own a particular y_i and is able to compute by himself a vector Z_i satisfying the ith equation, then he is able to solve the discrete logarithm problem of finding x_i.

The proof is now complete and the server has needed no particular previous information about user u. Also, the construction can be applied to a subset of the y_i's if the user does not wish to use all of them. □

When there are many users, say $u_0,\ldots,u_{m-1}$, instead of a single one, a way must be found to be able to grant the same right to more than one user, while keeping a unique y for it (the rights are user independent), no matter how many users have it. Algorithm 2 gives a solution to this problem. Let n_k be the number of rights to be granted to user u_k, for $k=0$ to $m-1$. Also, assuming that rights are granted first to u_0, then to u_1 and so on, let t_k be the number of rights, among the n_k to be granted to u_k, that have already been granted to some user in $\{u_0,\ldots,u_{k-1}\}$. Now A runs as follows:

Algorithm 2

For $k=0$ to $m-1$

1. Assume that the t_k rights having already been granted to someone else are $y_{k_0},\ldots,y_{k_{t_k-1}}$. Choose $n_k - t_k$ random integers x_{k_i}, $t_k \leqslant i \leqslant n_k - 1$ over $\mathcal{Z}/(p-1)$.
2. Pick a random vector $V_k = (a_{k_0},\ldots,a_{k_{t-1}})$ over $\mathcal{Z}/(p-1)$, such that all a_{k_j}'s are nonzero and no two among them are equal. a_{k_0} must be prime to $p-1$.
3. Find n_k vectors of size t, $Z_{k_i} = (z_{k_{i0}},\ldots,z_{k_{i(t-1)}})$, with nonzero coordinates, for $0 \leqslant i \leqslant n_k - 1$ over $\mathcal{Z}/(p-1)$, such that
$$\begin{aligned} x_{k_0} &= a_{k_0} z_{k_{00}} + \cdots + a_{k_{t-1}} z_{k_{0(t-1)}} \bmod (p-1) \\ &\vdots \\ x_{k_{n_k-1}} &= a_{k_0} z_{k_{(n_k-1)0}} + \cdots \\ &\qquad + a_{k_{t-1}} z_{k_{(n_k-1)(t-1)}} \bmod (p-1) \end{aligned} \tag{3}$$
To compute Z_{k_i} pick random nonzero $z_{k_{ij}}$, $j=1,\ldots,t-1$ over $\mathcal{Z}/(p-1)$ and solve the ith equation for $z_{k_{i0}}$ using that a_{k_0} can be inverted over $\mathcal{Z}/(p-1)$. If $z_{k_{i0}} = 0$, then repeat its computation with a new choice for the other $z_{k_{ij}}$'s.
4. Compute $y_{k_i} := \alpha^{x_{k_i}} \bmod p$, $t_k \leqslant i \leqslant n_k - 1$ and append them together with their meaning to the certified public rights list available to both servers and users.
5. Give the vectors Z_{k_i}, $0 \leqslant i \leqslant n_k - 1$ to u_k in cleartext (no need for certification).
6. Give the vector $V_k = (a_{k_0},\ldots,a_{k_{t-1}})$ to u_k in a confidential way.

It remains to be shown that it is possible to give *in the clear* a right $y_n = \alpha^{x_n} \bmod p$ to a user having rights $y_i = \alpha^{x_i} \bmod p$, $0 \leqslant i \leqslant n-1$ and a vector $V = (a_0,\ldots,a_{t-1})$. But this is straightforward since, following the procedure in step 3 of the previous algorithm, it is possible to compute some integers $z_{ni} \in \mathcal{Z}/(p-1)$, $0 \leqslant i \leqslant t-1$, such that
$$\begin{aligned} x_n = {} & a_0 z_{n0} + a_1 z_{n1} + \cdots \\ & + a_{t-1} z_{n(t-1)} \bmod (p-1). \end{aligned}$$
After this, the resulting vector Z_n is given *in cleartext* to the user and the procedure is finished. However, *the authority must take Subsection 2.1 into account when giving new rights to a user.*

Finally, this method has additional advantages concerning *rights revocation*. A way to revoke rights is for the authority A to publish a new certified rights list; then A also publishes the new vectors Z_{k_i} corresponding to the rights y_{k_i} which are maintained for each user u_k. Thus revocation can be viewed as granting rights again, but notice that this can be done in a *single public operation in the clear* (there is no need for confidential—i.e. encrypted—communication with each user whose rights are being revoked or maintained).

2.1. Collusion attacks and the security parameter t

If n rights have been granted to a user at the end of Algorithm 2, then assume linear independence for every t-subset among the Z_i, $i = 0,\ldots,n-1$ corresponding to those rights; this prevents using linear dependences in order for other users to guess a nonshared right x from a set (of less than t) shared ones. Now, it suffices for t to be greater than the number of rights granted to more than one user, in order to protect against *collusion* or *interuser attacks* (if a user or a group of users shared t rights with another user, they could determine the secret vector of the latter user and thus usurp the rest of his rights). Unshared rights do not matter. Let us justify the assumption on linear independence when $p \gg n,t$.

Theorem 2. *The number of $n \times t$ matrices Z over $\mathcal{Z}/(p)$, subject to: every choice of t rows is linearly independent, can be expressed by a monic polynomial of degree nt on p.*

Proof. Let Z_i, $i = 0,\ldots,n-1$ be the rows of Z. Z_0 can be arbitrarily chosen, i.e. we have $p^t - 1$ possibilities. Z_1 must be independent of Z_0, so we have $p^t - p$ possibilities for it. Z_2 cannot be in $\langle Z_0, Z_1\rangle$, so $p^t - p^2$ possibilities, and so on until Z_{t-1}, which cannot be in $\langle Z_0,\ldots,Z_{t-2}\rangle$ and offers $p^t - p^{t-1}$ possibilities. For $t \leqslant j \leqslant n-1$, Z_j cannot be in

$$\bigcup_{0 \leqslant i_0,\ldots,i_{t-2} \leqslant j-1} \langle Z_{i_0},\ldots,Z_{i_{t-2}}\rangle$$

and offers $p^t - \binom{j}{t-1}p^{t-1}$ possibilities. So the number of matrices we are looking for is

$$\prod_{j=0}^{t-1}(p^t - p^j)\prod_{j=t}^{n-1}\left(p^t - \binom{j}{t-1}p^{t-1}\right) = O(p^{nt}). \quad \square$$

Corollary 3. *The probability of choosing a random $n \times t$ matrix over $\mathcal{Z}/(p)$, subject to: every choice of t rows is linearly independent, approaches 1 as $p \to \infty$, for n and t fixed.*

Proof. Number of $n \times t$ matrices: p^{nt}. Number of $n \times t$ matrices fulfilling the above requirement: $O(p^{nt})$. $\square$

Note that the Z_i's of a user can be thought of as being completely random, in spite of (3), because the rights x_i are seen as random by the user. Choosing t according to the previous remarks completes the security for an isolated user shown in the proof of Theorem 1. For the rest, Algorithm 2 is an extension of Algorithm 1 and it is easy to see that it maintains the same structure for rights information. *Therefore, Theorem* 1 *can be restated with the same proof for Algorithm* 2 *and for every user $u \in \{u_0,\ldots,u_{m-1}\}$.*

3. Performance

The proposed scheme requires little authority computation, since Algorithm 2 involves only linear operations and no systems of equations are to be solved: users and rights can be efficiently managed. As for the user's rights possession proof to the server, it requires simultaneously proving knowledge of t constant logarithms in zero-knowledge with the [1] protocol, and also verifying that (2) holds. Both computations are fast.

As it has been shown, the scheme is very flexible, since user management can be done independently of servers and, thanks to the linear transformation (3), the secret pieces held by the user are constant and do not depend on the rights he owns at a given moment. Actually, it suffices for the user u_k to prove his identity V_k in order to use any subset of his rights, because V_k is the only secret parameter he holds.

As for the storage required, we have:

Authority

1. Secret storage for logarithms x_i.

2. Secret storage for all users' vectors V.
3. Read-write access to α, p and the list of the y_i's and their meanings (public certified data).

Servers
Read access to α, p and the list of the y_i's and their meanings (public certified data).

User u_k
1. Secret storage for his t-sized vector V_k (the protected read-only area of a smart card is a good place for V_k, because this vector has the advantage of being fixed).
2. Normal storage for his t-sized vectors Z_{k_i}, $0 \leqslant i \leqslant n_k - 1$.
3. Read access to α, p and the list of the y_i's and their meanings (public certified data).

Remark that no protected storage is needed at the servers.

Acknowledgment

Special thanks go to Professors Moncasi, Peralta and Rifà for their help in refining upon some topics discussed here.

References

[1] D. Chaum, J.-H. Evertse and J. van de Graaf, An improved protocol for demonstrating possession of discrete logarithms and some generalizations, in: D. Chaum and W.L. Price, eds., *Proc. Eurocrypt '87* (Springer, Berlin, 1988) 127-141.
[2] N. Koblitz, *A Course in Number Theory and Cryptography* (Springer, Berlin, 1987).
[3] S.C. Kothari, Generalized linear threshold scheme, in: G.R. Blakley and D. Chaum, eds., *Proc. of Crypto '84* (Springer, Berlin, 1985) 231-241.
[4] A. Shamir, How to share a secret, *Comm. ACM* **22** (1979) 612-613.