OPAL un système d'authentification par mots de passe non réutilisables

Nadia Nouali-Taboudjemat,
Mounir Benzaid, Bachir Mihoubi

Laboratoire Réseaux et Systèmes Répartis
Rue des trois frères Aissiou, Ben Aknoun,
E-mail : nnouali@wissal.cerist.dz

1. Introduction

La définition la plus large de l'authentification dans les systèmes informatiques englobe la vérification de l'identité, l'authentification de l'origine des messages et l'authentification du contenu des messages [**BRO94**]. Le concept de la vérification de l'identité est appliqué spécifiquement aux entités participant au traitement de l'information et ayant les capacités de la prise de décision, incluant les utilisateurs humains, les systèmes informatiques et les processus qui s'exécutent sur ces systèmes. D'un point de vue d'authentification, le terme "utilisateur" s'applique à toutes ces entités.

La méthode traditionnelle d'authentification des utilisateurs est basée sur des mots de passe secrets, qu'ils doivent soumettre quand ils demandent accès à un système particulier. La majorité des systèmes informatiques en utilisation de nos jours reposent sur les mots de passe pour l'authentification. L'avantage fondamental d'authentification basée seulement sur les mots de passe est qu'elle peut être implémentée entièrement par logiciel, évitant donc un coût de Hardware supplémentaire. Cependant, les responsables de la sécurité savent bien que la principale forme d'attaque contre les systèmes informatiques connectés à un réseaux tel que Internet, est l'espionnage des connexions réseaux dans le but de recueillir des identités et mots de passe d'utilisateurs légitimes. Ces identités et mots de passe usurpés système(par des méthodes de reniflage et de captage…) sont ultérieurement utilisés pour accéder au système.

Des techniques *d'authentification avancées,* telles que les cartes intelligentes, les jetons d'authentification, les méthodes biometriques et les méthodes logicielles, sont conçues pour palier aux faiblesses du système classique**[BRO94].** Malgré la diversité de ces nouvelles techniques, leur similitude réside dans le fait que les mots de passe qu'elles génèrent ne peuvent pas être réutilisés par un attaquant qui aurait espionné la connexion **[WAC94]**. En effet, le mot de passe n'est valable que pendant une seule session, et cela suppose la génération d'un nouveau mot de passe à chaque établissement de la connexion entre l'utilisateur et le système. L'authentification basée sur les mots de passe à usage unique(OTP) a été conçue dans ce contexte, son but est d'obliger l'utilisateur à s'identifier par un mot de passe différent à chaque ouverture de session. Ceci permet d'éviter que les mots de passe ne soient rejoués lors d'une attaque . Ce papier présente la mise en œuvre du logiciel ***OPAL***(One-time Password at Login) qui est un système de mots se passe à usage unique protégeant le système sécurisé des attaques externes dirigées vers son sous-système d'authentification.

2. Description du système *OPAL* [BEN 98)]

Le système *OPAL* et (OTP en général) permet de générer de multiples mots de passe à usage unique (les OTPs) à partir d'une seule phrase secrète appelée **"phrase de passe secrète"**(secret pass-phrase) et d'itérations récursives. Avec *OPAL, la phrase de passe,* connue de l'utilisateur seulement, ne traverse jamais le réseau, seuls les OTPs le traversent.
Le système *OPAL* fonctionne sur la base d'un challenge(défi) lancé à l'utilisateur. Le challenge est une chaîne de texte, à laquelle il ne peut correspondre qu'une seule réponse possible. Pour cela, l'idée d'utiliser des ***fonctions de hachage*** ou fonctions à sens unique(hash function ou One-Way function) pour générer des mots de passe à usage uniques est nécessaire. En effet, cette idée a été proposée pour la première fois par Leslie Lamport au début des années 80 puis mise en œuvre par le laboratoire Bellcore (Bell Communication Laboratory) sous la désignation de S/KEY One Time Password en 1991**[HAL94, RFC1704, RFC1760].** OTP a également été mis en œuvre par le laboratoire United States Naval Research et désigné par OPIE(OTP In Everything) **[SIY96].**

Il y a deux côtés participant dans l'opération du système *OPAL.* Du côté client(utilisateur), le *OTP* approprié doit être généré à partir de la phrase de passe secrète et de l'information fournie dans le défi du serveur. Du côté hôte, le serveur doit lancer le défi, qui inclut les paramètres de génération appropriés, au client; vérifier le *OTP* reçu; sauvegarder l'ancien *OTP* valide reçu et son numéro de séquence; et permettre aussi le changement sécurisé de la phrase de passe secrète de l'utilisateur.

Afin de générer un OTP, le client utilise une fonction de hachage à sens unique et itère cette fonction un nombre de fois précis en commençant par la valeur initiale (constituée de la phrase de passe secrète, et un germe reçu du serveur qui fait partie du défi) pour générer le *OTP*. Après chaque authentification réussie le nombre d'itérations diminue par un. Ainsi, une séquence unique des *OTPs* est générée. Le serveur vérifie le *OTP* reçu du client en lui appliquant une fois la même fonction et comparant le résultat obtenu avec l'ancien *OTP* valide. Chaque *OTP* peut être vérifié grâce à sa relation avec le *OTP* précédent, qui a lui-même été déjà vérifié.

2.1. Fonction de Hachage à Sens unique: « *Secure Hash Function* ou *One-Way Function* »

Une fonction de hachage à sens unique, $H(C)$, opère sur une chaîne de texte C de longueur arbitraire. Elle fournit une valeur de hachage h de longueur fixe beaucoup plus petite et vérifie les propriétés suivantes [**SCH97**, **CHA96**, **HAL94**]:

- Etant donné C , il est facile de calculer $h = H(C)$.
- Etant donné h , il est difficile de calculer C. C'est-à-dire l'entrée ne peut être regénérée à partir de la sortie; ce n'est donc pas un simple algorithme de compression ou de chiffrement
- Etant donné C, il est difficile de trouver une autre chaîne C' tel que $H(C) = H(C')$. C'est-à-dire la probabilité que deux entrées différentes (surtout de la même taille) produisent la même sortie est extraordinairement faible.

L'essence même des fonctions de hachage est de fournir une « **empreinte** » de C qui soit unique. Pour h donné, trouver un C' tel que $h = H(C')$ est extrêmement difficile. Idéalement, il ne devrait y avoir aucun moyen pour déterminer un tel C' autre que par essayer un nombre infaisable de valeurs et voire celle qui donne le bon résultat h. Si le nombre de valeurs

possibles de *C '* qui doivent être essayées est assez grand, alors pour tout objectif pratique la fonction ne peut être inversée.

Le système *OPAL* permet d'utiliser *MD4* et *MD5* (Message Digest Algorithm # 4 et # 5 : Algorithme du message abrégé). Conçu par Ronald Rivest de *RSA Data Security INC* [**RFC1320**], *MD4* accepte un nombre arbitraire de bits comme entrée et produit 16 octets (128 bits) de sortie. *MD4* est rapide, et il est connu pour sa sûreté [**HAL94**], c'est-à-dire il n'y a pas une méthode connue pour trouver l'entrée d'une sotie donnée qui est plus efficace que par les **essais exhaustifs** des entrées possibles. *MD5* qui est une version améliorée de *MD4* est similaire mais assure un niveau de sécurité plus élevé grâce à une étape de calcul supplémentaire [**SIY96**, **RFC1321**].

2.2. Déroulement de l'authentification OPAL

Le processus d'authentification ou d'ouverture de session des systèmes *Unix* standard (ou tout autre système multi–utilisateurs, par exemple : *VMS*) affichent une invite de login non sécurisée à laquelle l'utilisateur répond par son mot de passe. Si l'*UID* (*USER ID*) et le mot de passe correspondent, l'utilisateur est accepté et sera autorisé à accéder au système.
Avec *OPAL*, l'utilisateur doit répondre à un défi : il est sensé répondre à une question dont il est le seul à connaître la réponse. A la différence avec l'authentification Unix qui se réalise en une seule étape (entrée d'un password secret au prompt **password :**), *OPAL* suppose qu'une opération de génération (avec une calculette logicielle *opalkey*, *winkey*, voir figure 2) aura été lancée auparavant à partir d'un terminal sécurisé, ou bien que l'utilisateur aura accès à un calculateur sur sa machine en local. On peut envisager une simplification de la séquence au moment de la connexion en ayant générée à l'avance l'*OTP* correspondant à la séquence. Comme le montre la figure 1, le processus d'authentification est le suivant :

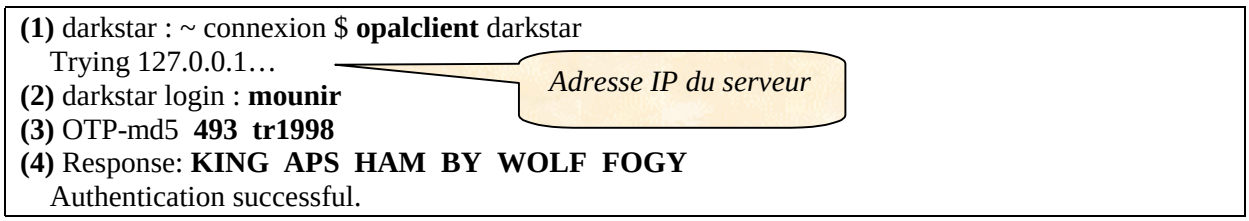


Figure 1. Le schéma d'authentification *OPAL* .

(1) Quand le client se connecte au serveur "darkstar", ce dernier lui demande son nom de login.

(2) Le client fournit son nom de login " mounir ".

(3) Le serveur recherche le nom de login dans sa base de données */etc/opalkeys* et lance le défi approprié qui se compose du nombre d'itération " 493 " du *OTP* attendu et la valeur de germe "tr1998" extraits de la base de données. Cette dernière est la base dans laquelle sont stockées les informations du système *OPAL*, contenant essentiellement pour chaque utilisateur : le nom de *login*, le nombre d'itération, le germe et le dernier OTP valide dans la séquence. La forme standard du défi est [**RFC1938**] :

OTP-< algorithm identifier > < sequence integer > < seed >

– sequence integer : le nombre d'itération ou numéro de séquence de l'*OTP* (nombre d'application de la fonction de hachage) .

– seed : le germe est utilisé pour permettre au client d'utiliser la même phrase de passe secrète sur plusieurs machines à partir desquelles il prévoit de se connecter (en utilisant des germes différents); ou de reprendre systématiquement la même phrase de passe secrète pour toutes ces initialisations sur la même machine, en changeant seulement le germe. Comme pour chaque machine le germe est différent, la combinaison avec la phrase de passe secrète et le nombre d'itération initialisé sera elle aussi différente. Le germe sera constant jusqu'à l'exécution ultérieure d'une nouvelle initialisation .

(4) Le client fait appel à son calculateur pour générer la réponse au défi (l'*OTP*) à partir des valeurs du nombre d'itération et du germe présentés par le prompt *OPAL*, mises en relation avec la phrase de passe secrète connu seulement de l'utilisateur, en utilisant l'algorithme de hachage à sens unique (*MD5* ou *MD4*) autant de fois que le nombre d'itération.

Etape de Calcul : A l'étape initiale, la phrase de passe secrète est concaténée à un germe. Le résultat de cette concaténation est passé par la fonction de hashage. La séquence de génération des *OTP* commence par l'*OTP* de nombre d'itération **" N "**(par défaut 499), tel que N est configuré pour l'utilisateur lors de son ajout à *opalkeys.* Pour avoir l'*OTP* d'itération N, il faut appliquer N fois le *MD5* ou *MD4* au résultat S de l'étape initiale. A chaque authentification réussie, la valeur d'itération est décrémentée d'une unité et c'est ce qui va permettre l'utilisation d'un mot de passe différent à chaque *login*. Le serveur sauvegarde à l'initialisation l'*OTP* d'itération N et demande au client l'*OTP* d'itération N-1, ce qui permet de vérifier la validité de l'*OTP* d'itération N-1 en appliquant *MD5* ou *MD4* sur l'*OTP* d'itération N-1, qui devrait donner l'*OTP* d'itération N. De cette façon, le serveur sauvegarde à chaque fois l'ancien OTP_i (*OTP* d'itération i) et demande l'OTP_{i-1} au client, et puisque *MD5* ou *MD4* n'est pas convertible un renifleur ne pourra pas obtenir la valeur du prochain OTP_{i-1} à partir de la valeur courante d'OTP_i .

Une fois la réponse générée " KING APS HAM BY WOLF FOGY ", il la rend au serveur**[RFC1760, RFC1938**].

(5) Le serveur applique *MD5* ou *MD4* une fois sur l'*OTP* d'itération 493 reçu, et compare le résultat avec l'ancien *OTP* d'itération 494 sauvegardé dans *opalkeys* pour ce nom de login; s'ils sont égaux alors l'utilisateur est déclaré authentifié et sera accepté par le système.
Le nombre d'itération et le germe sont spécifiques à l'utilisateur et ont été configurés lors de l'ajout de l'utilisateur à *opalkeys*. Il est à noter que, même si deux utilisateurs prennent les mêmes valeurs de nombre d'itération et de germe, les mots de passe générés seront différents s'ils ont choisi des phrases de passe différentes.

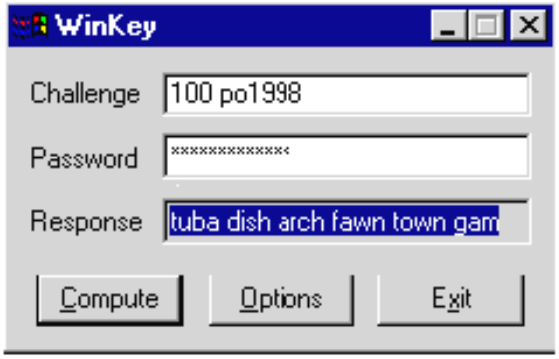
3. Composantes logicielles du système

L'implémentation du système d'authentification *OPAL* comprend plusieurs éléments :

3.1 - Outils de génération de mots de passe *OTP*

Des outils de génération de mots de passe à usage unique OTPs pour le système d'exploitation Linux et d'autres plates-forme, sont mis à la disposition des utilisateurs . **La commande *OPALKEY*** n'est que l'un des nombreux noms du même programme connu également sous les dénominations ***opal-md4, opal-md5, otp-md4 ou otp-md5***. ***OPALKEY*** permet aussi de générer une liste d'OTPs à l'avance pour travailler sur un poste démuni de calculateur. **La commande *OPALGEN*** est un simple générateur pouvant être utilisé à la place de *opalkey*. **La commande *WINKEY*** [**AYL96**] est un calculateur *OTP* de Bellcore *S/Key* et *NRL OPIE* pour *MS-Windows* supportant *MD4* et *MD5*, compatible avec *opalkey*,

avec les renforcements de l'interface utilisateur graphique. Le but de ce calculateur est de minimiser le nombre des étapes supplémentaires impliquées par la séquence d'authentification (*défi / réponse*). Pour cela, il dispose de mécanisme *couper-coller* automatique. Il suffit de sélectionner le challenge en le noircissant avec la souris dans la fenêtre dans laquelle s'effectue la connexion (par exemple, *Telnet* sur *windows*). Puis de déplacer le pointeur de la souris dans la fenêtre" Challenge " de *winkey*, pour que le challenge soit copié dans cette fenêtre. De plus, le *OTP* produit par le calculateur est automatiquement sélectionné. Cette commande est fournie avec le système *Windows* et nous n'avions pas à l'implémenter.W*inkey32* est pour *windows95* ou *NT*, et *winkey* pour *windows3.x*. La fenêtre *winkey* est représentée dans la figure 2, elle est utilisée pour générer à l'avance des *OTP*, et peut être configurée de façon à lire le défi depuis le presse-papier (*clipboard*) et copier la



valeur calculée dans ce dernier .

Figure 2. Fenêtre *winkey*.

3.2. Outil d'initialisation :

Un outil d'initialisation et de mise à jour de la base de données /etc/opalkeys sous linux, est mis à la disposition de l'administrateur système. Cet outil est **la commande *OPALINIT*** qui modifie ou établie un mot de passe utilisateur pour le système d'authentification OPAL .

Pour initialiser un mot de passe OPAL, il faut que l'utilisateur soit déjà connecté sur la machine sur laquelle il désire initialiser le processus d'authentification par *OPAL*. Seulement le *root* (super-utilisateur) a le droit de changer les mots de passe *OPAL* d'autres utilisateurs, c'est-à-dire l'utilisateur ne peut initialiser que son mot de passe. Il faut disposer localement d'un générateur d'*OTP*, en cas d'initialisation à distance, car seul un mot de passe à usage unique sera passé sur le réseau.

En cas de réinitialisation d'un mot de passe *OPAL*, ***OPALINIT*** fonctionne comme la commande ***Passwd*** d'*Unix*. C'est-à-dire qu'elle demande à l'utilisateur de taper son dernier mot de passe *OPAL*, en utilisant l'ancienne phrase de passe secrète avant de rentrer le nouveau. La nouvelle phrase de passe secrète est demandée deux fois afin de se protéger contre les erreurs de frappe .

Il est bien entendu que ces mots de passe sont fournis sous forme d'*OTPs*, ainsi il est complètement sécurisé de changer de mot de passe à travers le réseau.

Exemples :

***Utiliser Opalinit à partir de la console* :**

Systeme $ **opalpinit - c**
Updating **bachir** :
Reminder : Only use this method from console;
NEVER from remote. If you are using telnet, xterm or a dial-in,
type ^C now or exit with no password.
then run opalinit without - c parameter.
Using *MD5* to compute responses.
Old secret password : **********
New secret password : **********
New secret password (again) : **********

ID travail OPAL key is **499 be93564**
HAW DIAL NOOK MEW BUNT HALE
Systeme $

L'utilisateur qui a activé la commande.

Les valeurs par défaut, car les options - **n**, - **s** ne sont pas précisées.

Converti en hexadécimale, et stocké dans ***opalkeys***.

***Utiliser Opalpinit à distance* :**

Comme le montre la figure 3, le programme *opalinit* fait authentifier l'utilisateur par le dernier *OTP* généré de la séquence, en utilisant l'ancienne valeur de la phrase de passe secrète et les arguments de défi associées " 497 se1975 ", avant d'afficher l'invite contenant l'itération et le germe configurés selon l'utilisateur par les options " -n 100 " et " -S se1974 ". Pour mener à son terme le processus, l'utilisateur aura accès à un calculateur *OTP* sur sa machine en local (comme dans cet exemple " *winkey* "), ou bien qu'une commande de génération (*opalkey*, *opalgen*) aura été lancée auparavant à
partir d'un terminal sécurisé. L'utilisateur calcule l'*OTP* correspondant en utilisant une nouvelle phrase de passe, s'il le souhaite, et le transmet au système hôte.

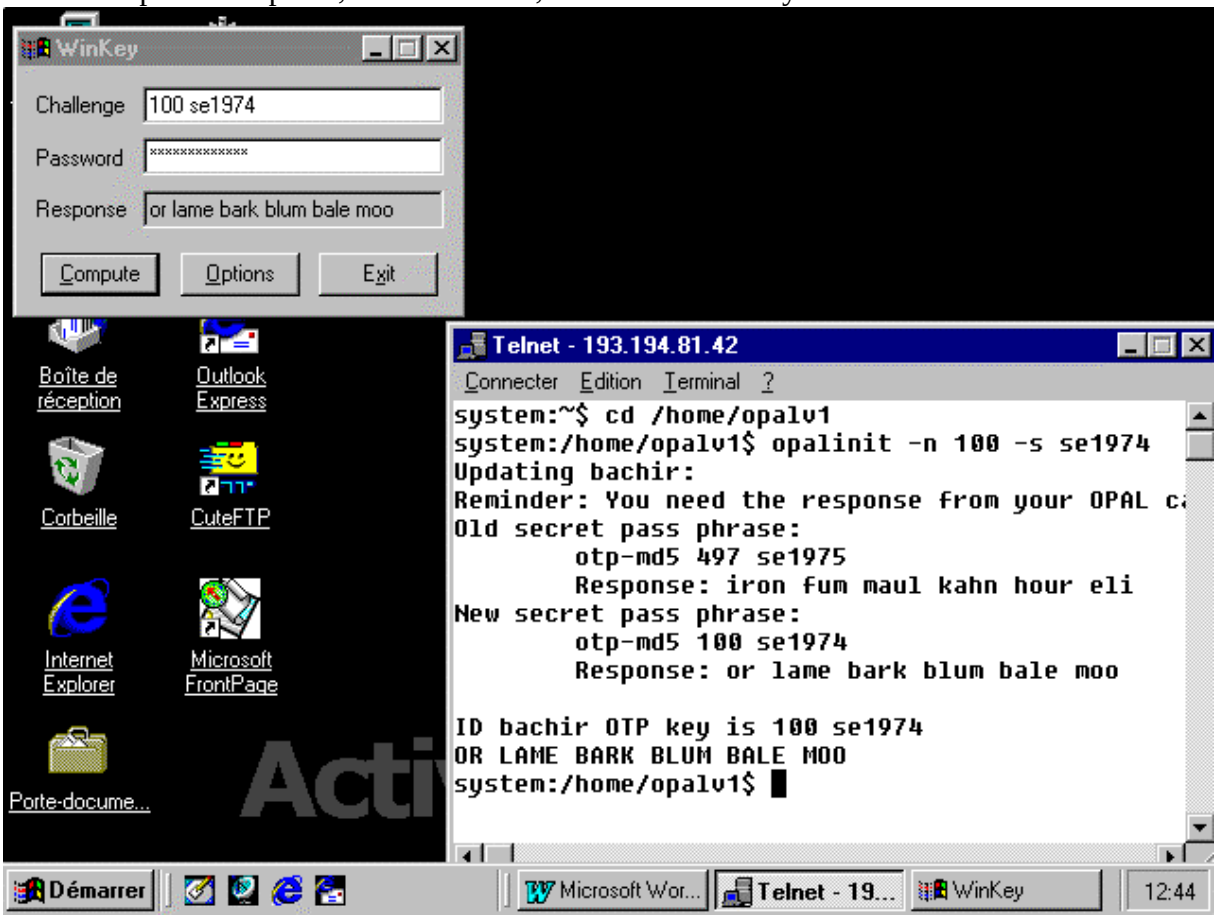


Figure 3. Initialisation d'un utilisateur à distance

Ainsi, même en présence d'une oreille indiscrète l'initialisation peut se faire sans risque à travers le réseau, car la valeur secrète ne traverse jamais le réseau, mais contribue au calcul local de l'*OTP* de la nouvelle séquence qui sera autorisé à se déplacer sur le réseau et suffit au système hôte pour conduire des authentifications ultérieures .

4. Applications pratiques

Le système réalisé a été expérimenté sur des applications de connexion à distance(login à distance, Telnet et FTP). Pour ces applications la phase d'authentification classique a été remplacée par une phase de négociation d'un challenge *OPAL*. Voici l'exemple de TELNET.
Les modifications se rapportent seulement au niveau du serveur, puisque c'est lui le responsable de l'authentification *OPAL*, et la base de données /etc/opalkeys réside de son côté. Le client *Telnet*, qui peut être sur n'importe quel plate-forme (dans ce cas W*indowsNT*), se contente de créer une connexion TCP avec le serveur, accepte les sorties provenant du serveur (par exemple : " login : ", < défi >, " Response ou Password : ") et les affichent à l'utilisateur, et envoie les entrées de ce dernier au serveur .

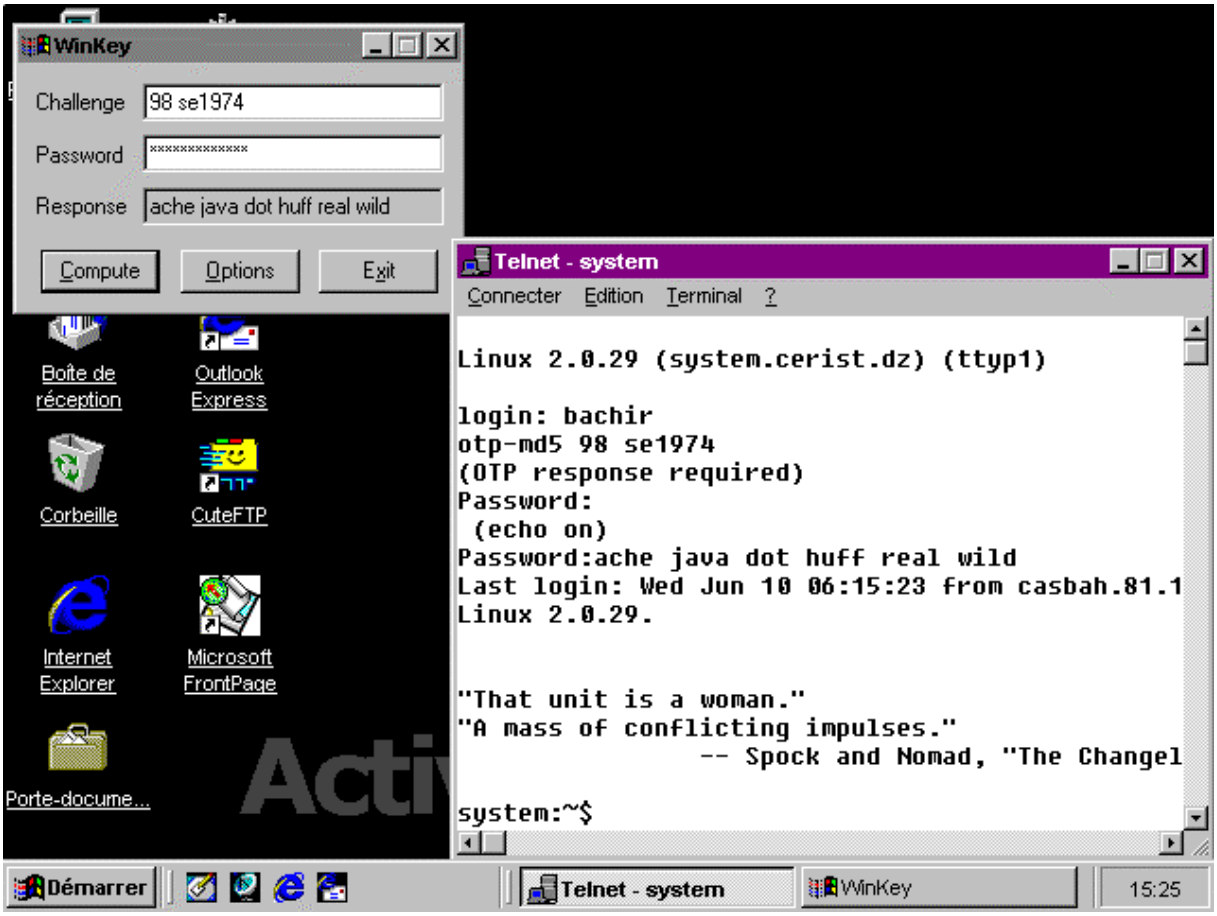


Figure 4. Ouverture de session *Telnet* avec des défis *OPAL*.

Pour une meilleure structuration du système *OPAL* qui englobe plusieurs tâches : calcul, authentification, initialisation …, ses différentes fonctions ont été regroupées dans une bibliothèque **libopal.a**, rendant ainsi la modification et l'écriture d'autres programmes plus

facile. La bibliothèque peut être utilisée par tous les programmes objets du système (client, serveur, …)

5. Conclusion

Les performances du système d'authentification *OPAL* tel qu'il est implémenté dépendent des performances des applications l'utilisant(Login, Telnet,…). Les programmes *login* et *Telnet* existent sur *Unix* et Internet, et donc le seul facteur qui peut influer sur leurs performances est le temps supplémentaire qui est ajouté par la phase de génération de l'OTP. L'utilisation des calculateurs au moment de la connexion, (par exemple winkey pour le système *windows*, avec le mécanisme *couper-coller*) ou une liste des *OTP* générés d'avance, ou tout autre calculateur externe augmente le temps de l'ouverture de session de quelques secondes par rapport au temps de login *Unix* classique.

Nos calculateurs *Linux* : *opalkey* et *opalgen* sont rapides et fiables. La commande *opalinit* facilite l'administration du système et la mise en pratique de *OPAL* par la configuration des utilisateurs du système. Elle fonctionne comme la commande ***Passwd*** de *Unix* pour le changement de mot de passe, son point fort est le changement de la phrase de passe secrète a travers le réseau sans risque d'écoutes passives. Divers contrôles de sécurité sont fournies et contribuent au renforcement de protection du site on peut citer : la taille minimale de la phrase de passe secrète et le germe, l'interdiction de passage de la phrase de passe secrète à travers le réseau, l'interdiction de mise à jour de la base de données *opalkeys* par un utilisateur qui veut initialiser le compte d'un autre utilisateur, sauf le super utilisateur qui a ce droit, et la vérification de l'intégrité de l'*OTP*, par l'ajout de 2 bits supplémentaires (checksum) à l'*OTP*, et qui sera contrôlé au niveau du serveur .

OPAL n'est cependant pas une solution idéale, puisqu' elle protège contre les attaques passives et ne protège pas contre les attaques actives et le détournement de connexion après que l'utilisateur se soit authentifié auprès du système. Le niveau de sécurité offert est plus bas que celui offert par *KERBEROS* qui est un système assurant la sécurité durant toute la session **[BID95, HAL98]**. La sécurité du système *OPAL* dépend de la force cryptographique de la fonction de hachage, et tant que cette dernière n'est pas vulnérable, le système reste fiable. Mais actuellement, une version améliorée et plus efficace de *MD5* est utilisée. Dans la cryptographie, il existe une dizaine de fonctions on peut citer *MD2*, *RIPE-MD*, *HAVAL*, *GOST* et *SHA*. D'après Bruce Shneier [**SCH97**], *SHA* est plus efficace. *SHA* ressemble à *MD5* par l'addition d'une étape de calcul supplémentaire et un meilleur effet d'avalanche. Et comme il produit une empreinte de 160 bits, il est plus résistant à une attaque exhaustive que les autres algorithmes. La condensation de l'empreinte à 64 bits nécessaire pour l'*OTP* est facile à faire.

Références BIBLIOGRAPHIES

[**AYL96**] : David Aylesworth, " winkey pour les utilisateurs de l'Irisa ", Technologic. Inc – http://www.tlogic.com/, 1996.

[**BEN98**] : Mounir Benzaid, Bachir Mihoubi, Nadia Nouali-Taboudjemat**,** "Mise en œuvre d'un système d'authentification avancée", Mémoire réalisé au Cerist(Centre de Recherche sur l'Information scientifique et Technique, N°55/98, 1998.

[**BID95**] : Christophe Bidan & Valérie Issarny, " Un Aperçu des Problèmes de Sécurité dans les Systèmes Informatiques ", Institut de Recherche en Informatique et Systèmes Aléatoires (*IRISA*)-Publication Interne N° 959, Octobre 1995, France.

[**BRO94**] : Ronald H. Brown, Arati Prabhakar, " Guideline for the use of Advanced Authentication Technologie Alternatives ", Federal Information Processing Standard Publication 190, National Institue of Standards and Technology (*NIST*), septembre 1994, USA.

[**CHA96**] : D. Brent Chapman, Elizabeth D. Zwicky, " Firewalls la Sécurité sur Internet ", Traduction de Jean Zundel, Edition O'Reilly International Thomson, Paris 1996.

[**GRE98**] : GRECC-LIENS - Ecole Normale Supérieure, Pages HTML : " Authentification avec *Kerberos* ", wwwgrecc@dmi.ens.fr, 1998.

[**HAL94**] : Neil M. Haller, " The S/KEY One-Time Password System ", Bellcore Morristown, New Jersey, Proceedings of the ISOC Symposium on Network and Distributed System Security, février 1994, San Diego, CA.

[**RFC132**] : Ron Rivest, " The *MD4* Message-Digest Algorithm ", MIT and RSA Data Security, Inc, April 1992.

[**RFC132**] : Ron Rivest, " The *MD5* Message-Digest Algorithm ", MIT and RSA Data Security, Inc, April 1992.

[**RFC1704**] : N. Haller, R. Atkinson, " On Internet Authentication ", Bell Communications Research and Naval Research Laboratory, october 1994.

[**RFC1760**] : Neil. Haller, " The S/KEY One-Time Password System", Bellcore, February 1995.

[**RFC1938**] : Neil Haller (Bellcore), Craig Metz (Kaman Sciences Corporation), " A One-Time Password System ", May 1996.

[**SCH97**] : Bruce Scheneier , " Cryptographie Appliquée : Algorithmes, protocoles et codes source en C ", Traduction de Laurent Viennot, 2° édition International Thomson Publishing France, Paris 1997, ISBN : 2-84180-036-9.

[**SIY96**] : Karanjit Siyan & Chris Hare, INTERNET Sécurité & Firewalls, Traduit de l'américain par Veronique Campillo, Publié par Simon & Schuster Macmillan, Novembre 1996(France), ISBN : 2674400-01996-1.

[**WAC94**] : John P. Wack, Lisa J.Carnahan, " Keeping Your Site Comfortably Secure : An Introduction to Internet Firewalls ", National Institute of Standards and Technology (NIST) Special Publication 800-10, U.S. Departement of Commerce .