HAL
open science

JRES 2005 : La mémorisation des mots de passe dans les navigateurs web modernes

Didier Chassignol, Frédéric Giquel

To cite this version:

Didier Chassignol, Frédéric Giquel. JRES 2005 : La mémorisation des mots de passe dans les navigateurs web modernes. JRES 2005 : La mémorisation des mots de passe dans les navigateurs web modernes, Dec 2005, Marseille. inria-00000404v2

HAL Id: inria-00000404

https://inria.hal.science/inria-00000404v2

Submitted on 21 Oct 2005

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

La mémorisation des mots de passe dans les navigateurs web modernes

Didier Chassignol
INRIA
Didier.Chassignol@inria.fr

Frédéric Giquel
INRIA
Frederic.Giquel@inria.fr

Résumé

Les navigateurs web modernes proposent, lors d'une authentification, de mémoriser le mot de passe utilisé. On peut se demander si cette fonctionnalité de mémorisation peut être utilisée pour simplifier la vie de l'utilisateur tout en assurant un niveau de sécurité satisfaisant. Cet article étudie cette fonctionnalité pour les principaux navigateurs (Mozilla, Netscape, Firefox, Internet Explorer, Safari et Konqueror) ainsi que les mécanismes sous-jacents.

Nous rappellerons les principales méthodes d'authentification des utilisateurs existantes sur les serveurs web. Nous présenterons les tests effectués sur les navigateurs ainsi que les résultats associés suivant plusieurs catégories : disponibilité de la fonctionnalité de mémorisation, ergonomie, sécurité des données mémorisées sur le poste client et sécurité des échanges.

Enfin nous verrons si l'utilisation de ces fonctionnalités est appropriée dans deux contextes d'utilisation différents (applications web utilisant une base de comptes commune ; ensemble hétérogène d'applications web) et proposerons des recommandations d'utilisation et de configuration par navigateur.

Mots clefs

Authentification, SSO, stockage sécurisé, applications web, navigateurs web

1 Introduction

Les utilisateurs sont confrontés à un nombre croissant d'applications web nécessitant de s'authentifier et doivent donc effectuer de multiples saisies de mot de passe dans une même session de travail.

Dans le but de cacher ces authentifications à l'utilisateur, les navigateurs web modernes proposent des fonctionnalités de mémorisation des mots de passe.

On peut alors se demander si cette fonctionnalité des navigateurs peut être utilisée pour simplifier la vie de l'utilisateur tout en assurant un niveau de sécurité satisfaisant.

2 Méthodes d'authentification des utilisateurs sur les serveurs web

Les méthodes d'authentification décrites dans cette section sont indépendantes de la manière dont est stocké le mot de passe sur le serveur (dans un fichier, une base de données ou un annuaire LDAP par exemple). Elles se différencient par la façon dont le navigateur et le serveur web communiquent.

2.1 Authentification au niveau du protocole HTTP

Il existe deux méthodes d'authentification par login/mot de passe au niveau du protocole HTTP : Basic et Digest. Ces deux méthodes sont normalisées et définies dans la RFC 2617 [1].

L'utilisation d'une authentification intégrée au protocole HTTP a principalement deux conséquences :
- Le navigateur web peut facilement savoir si l'authentification a réussi ou échoué (grâce à des codes de retour normalisés) et donc détecter si un mot de passe n'est pas valide.
- Comme HTTP est un protocole sans état, les requêtes sont dissociées et l'authentification doit donc être rejouée à chaque fois. Pour implémenter un suivi de session authentifiée et ainsi éviter que l'utilisateur n'ait à ressaisir son mot de passe pour visualiser chaque page, les navigateurs mémorisent celui-ci (en mémoire vive) pour rejouer l'authentification automatiquement. Ce rejeu automatique n'est valable que pour les pages du même serveur qui utilisent un identifiant commun appelé « realm ».

Lorsque l'accès à un serveur web nécessite une authentification, le navigateur fait apparaître un pop-up dans lequel l'utilisateur doit remplir les entrées « Utilisateur » et « Mot de passe ».

Les différences entre les méthodes Basic et Digest apparaissent dans les informations échangées sur le réseau et ne sont pas visibles par l'utilisateur.

Authentification Basic. Le mécanisme utilisé se déroule en plusieurs étapes :

1. Le navigateur web tente d'accéder à une page protégée

par l'authentification Basic sans fournir de login/mot de passe ;

2. Le serveur répond avec le code d'erreur 401 `Unauthorized` et en spécifiant un « realm » ;
3. Le navigateur web retente d'accéder à la page protégée en fournissant un couple login/mot de passe (dans une chaîne de caractères codée en base64[1]) qui provient :
   – si possible, de la mémoire du navigateur et correspondant au « realm » spécifié par le serveur et au nom du site web ;
   – ou à défaut, de l'utilisateur à travers un pop-up ;
4. Si le login/mot de passe correspond à celui attendu par le serveur, la page demandée est fournie ; sinon, le serveur retourne le code d'erreur 401 et le navigateur web doit fournir un autre couple login/mot de passe (qu'il redemande généralement à l'utilisateur).

Authentification Digest. Ce mécanisme est une authentification de type « challenge/réponse » qui permet d'éviter la transmission en clair du mot de passe sur le réseau et qui empêche (dans une certaine mesure) les attaques par rejeu.

Cette méthode d'authentification est très rarement utilisée car son implémentation dans Internet Explorer ne respecte pas entièrement la RFC et est incompatible avec le serveur web Apache. Par conséquent ce type d'authentification n'est pas pris en compte dans la suite de l'article.

2.2 Authentification applicative par formulaire

Plutôt que de se baser sur une authentification normalisée au niveau HTTP, beaucoup d'applications web utilisent leur propre système d'authentification basé sur un formulaire dans une page web. Pour s'authentifier, l'utilisateur doit alors remplir deux champs qui sont généralement « Utilisateur » ou « Identifiant » (entrée de formulaire de type `text`) et « Mot de passe » (entrée de formulaire de type `password`).

Le mécanisme généralement utilisé. Exemple d'authentification réussie :

1. Le login et le mot de passe sont envoyés au serveur (par la méthode POST ou parfois GET) ;
2. Le serveur vérifie la validité des authentifiants et crée un identifiant unique de session ;
3. Le serveur envoie au client un cookie contenant l'identifiant de session ;
4. Le navigateur web stocke le cookie et réutilise l'identifiant de session pour s'authentifier à chaque nouvelle requête.

Toutes les transmissions de données sont généralement en clair. Dans ce mécanisme, le cookie est un élément important même si il n'est pas utilisé directement pour l'authentification car il permet d'implémenter un suivi de session authentifiée. L'identifiant de session qu'il contient doit donc être gardé confidentiel pendant sa durée de validité (qui est généralement limitée à quelques minutes ou quelques heures par le serveur).

L'utilisation d'un cookie pour implémenter un suivi de session HTTP est documentée dans la RFC 2965 [2].

Détection par le navigateur. L'authentification applicative par formulaire n'étant pas normalisée, le navigateur ne peut la détecter que partiellement.

En effet, le navigateur est capable de détecter la présence d'une authentification par formulaire (grâce à la présence d'une entrée de formulaire de type `password` dans la page web) mais ne peut pas savoir si l'authentification a réussi ou échoué. En conséquence, en cas d'échec de l'authentification, c'est l'application ou l'utilisateur qui aura la charge de renégocier une authentification.

Stockage des cookies. La technique de stockage du cookie dépend des informations sur sa durée de validité fournies par le serveur :
– Si le cookie est valide uniquement pour la session, le navigateur le garde en mémoire sans l'écrire sur le disque dur.
– Si le cookie est valide jusqu'à une date précise, le navigateur l'écrit sur le disque dur.

2.3 Les apports de HTTPS

Les méthodes décrites ci-dessus ne concernent que l'authentification de l'utilisateur. L'utilisation conjointe de HTTPS apporte une meilleure sécurité en permettant :
– l'authentification du serveur (grâce à un certificat et une autorité de certification reconnue par le navigateur) ;
– le chiffrement de la communication.
HTTPS peut aussi permettre d'authentifier l'utilisateur grâce à un certificat client. Cette méthode d'authentification ne se basant pas sur l'utilisation de login/mot de passe, elle sort du cadre de notre étude.

3 Points étudiés

Les fonctionnalités de mémorisation des mots de passe ne sont pas normalisées. Par conséquent, leur implémentation est différente dans chaque navigateur.

Nous avons étudié ces fonctionnalités pour les principaux navigateurs web utilisés à l'INRIA et plus généralement dans la communauté éducation/recherche :
– Mozilla 1.7.3, Netscape 7.1 et Firefox 1.0 sous Windows, Linux et Mac OS X
– Internet Explorer 6.0 sous Windows XP SP2
– Internet Explorer 5.2.3 sous Mac OS X
– Safari 1.2.4 sous Mac OS X 10.3
– Konqueror 3.4.2 sous Linux

[1] Base64 étant simplement une modification du codage, la transmission du mot de passe n'est pas confidentielle.

Pour chaque navigateur, en nous basant sur la documentation disponible et sur des tests d'accès à plusieurs applications web, nous avons tenté de répondre aux questions détaillées dans les quatre sous-sections suivantes.

3.1 Disponibilité de la fonctionnalité de mémorisation

- Le navigateur dispose-t-il de la fonctionnalité de mémorisation des mots de passe ?
- Fonctionne-t-elle pour les deux catégories d'authentification ? En HTTP et HTTPS ?

3.2 Ergonomie

- Comment l'utilisateur peut-il contrôler la mémorisation d'un mot de passe ?
- Comment l'authentification est-elle automatisée ?
- Comment s'effectue un changement de mot de passe ?
- Quel est le comportement lorsque plusieurs applications sont sur le même site web ? La mémorisation est-elle globale au site web ?
- Peut-on regrouper des mots de passe communs à plusieurs applications pour les gérer de manière unifiée ?
- Peut-on visualiser les mots de passe mémorisés ?

3.3 Sécurité des données mémorisées sur le poste client

Nous avons vérifié si les données d'authentification sur le poste client sont sécurisées en cas d'accès physique (vol de portable) ou logique (accès au système de fichier ou à la mémoire d'un système en fonctionnement) :

- Où sont stockés les mots de passe sur le poste client ?
- Le stockage est-il sécurisé ? Est-il protégé par un mot de passe (appelé « master password ») ?
- Où et comment sont stockés les cookies ?
- Peut-on utiliser une solution de chiffrement annexe (EFS pour Windows, CryptoAPI pour Linux, ...) afin de sécuriser le stockage ?
- Peut-on sauvegarder puis restaurer le stockage ?

3.4 Sécurité des échanges

Pour que l'échange soit sécurisé, la communication doit être chiffrée et le serveur authentifié. Ceci étant réalisé lorsque HTTPS est utilisé, les points étudiés sont :

- Peut-on limiter l'usage de la mémorisation des données d'authentification aux seules pages HTTPS ?
- Lorsque des données sont mémorisées pour une page HTTPS, peut-on être sûr que le navigateur ne les utilisera que pour le site web légitime et uniquement en HTTPS ?

4 Principaux résultats des tests

4.1 Disponibilité de la fonctionnalité de mémorisation

	Authentification HTTP Basic		Authentification applicative par formulaire	
	HTTP	HTTPS	HTTP	HTTPS
Mozilla, Netscape et Firefox	oui	oui	oui	oui
Internet Explorer sous Windows	oui	oui	oui	oui
Internet Explorer sous Mac OS X	oui	oui	non	non
Safari	oui	oui	oui	oui
Konqueror	oui	oui	oui	oui

4.2 Ergonomie

Contrôle de la mémorisation par l'utilisateur. Pour tout les produits testés, la mémorisation d'un mot de passe est contrôlée au cas par cas au moment de l'authentification.

Dans le cas de l'authentification HTTP Basic, le navigateur propose une case à cocher dans le pop-up d'authentification.

Dans le cas de l'authentification applicative par formulaire, lorsque le navigateur détecte celle-ci, une fenêtre apparaît pour demander à l'utilisateur si il veut mémoriser le mot de passe.

Automatisation de l'authentification. Pour les navigateurs permettant l'utilisation d'un « master password » (Mozilla, Netscape, Firefox, Safari et Konqueror ; voir « Stockage des mots de passe » dans la section 4.3), celui-ci doit être fourni par l'utilisateur :

- lors de la première réutilisation, dans une session du navigateur, d'un mot de passe mémorisé ;
- lorsque sa durée de vie a expiré (voir « Durée de vie du « master password » » dans la section 4.3).

Lorsque le « master password » est fourni, l'authentification n'est pas entièrement automatisée car l'utilisateur doit valider un formulaire ou un pop-up prérempli, à l'exception de l'authentification HTTP Basic avec Safari. On peut noter que cette automatisation totale dans Safari entraîne un risque de rejeu et de vol, à l'insu de l'utilisateur, des mots de passe mémorisés dans les pages n'utilisant pas HTTPS.

Notons pour finir deux éléments limitant l'automatisation de l'authentification dans le cas de l'authentification applicative par formulaire :

- Si le formulaire d'authentification propose un choix dans une liste déroulante[2], aucun navigateur ne mémorise la valeur choisie par l'utilisateur dans cette liste. L'authentification est alors simplifiée mais pas entièrement automatisée.

[2] Par exemple, le formulaire d'authentification du logiciel de webmail Horde IMP permet de choisir la langue dans une liste déroulante.

– Le serveur web peut demander au navigateur de ne pas mémoriser tout ou partie des champs d'un formulaire (y compris les mots de passe) en utilisant la valeur `off` pour l'attribut `autocomplete` de l'entrée concernant le mot de passe dans le formulaire [3]. Cette « sécurité » peut être contournée, mais pas facilement par un utilisateur standard.

Changement de mot de passe. Le changement de mot de passe s'effectue :
– soit par un formulaire de changement de mot de passe (y compris pour l'authentification HTTP Basic) ;
– soit directement par un administrateur sur le serveur.
Pour ces deux possibilités, le changement de mot de passe n'est pas détecté par le navigateur qui continue de mémoriser l'ancien mot de passe. La mémorisation du nouveau mot de passe se déroule lors de la première authentification après le changement de mot de passe.

Dans le cas de l'authentification HTTP Basic, après le changement du mot de passe, le navigateur va détecter un échec d'authentification et proposer un nouveau pop-up d'authentification. On constate alors de très légères différences entre les navigateurs qui selon les cas proposeront soit un pop-up entièrement vierge, soit un pop-up plus ou moins prérempli.

Dans le cas de l'authentification applicative par formulaire, le formulaire est prérempli avec l'ancien mot de passe. Si l'utilisateur valide cette proposition, il obtient en retour la page d'échec d'authentification du site. Si l'utilisateur modifie le mot de passe prérempli, le navigateur prévient que le mot de passe mémorisé est différent et propose à l'utilisateur de remplacer l'ancien mot de passe.

Plusieurs applications sur un même site web. Dans le cas de l'authentification HTTP Basic, les navigateurs Internet Explorer sous Windows, Firefox, Netscape et Safari mémorisent le « realm » associé à l'authentification et peuvent donc automatiser correctement l'authentification lorsqu'il y a plusieurs applications sur le même site web. Avec les autres navigateurs, le mot de passe proposé est le dernier utilisé sur le site web et l'utilisateur doit donc modifier la valeur préremplie ; l'automatisation n'est donc pas possible et l'ergonomie n'est pas satisfaisante.

Dans le cas de l'authentification applicative par formulaire, l'URL de la page contenant le formulaire est mémorisée avec le mot de passe pour Internet Explorer sous Windows et Konqueror. Ces deux navigateurs gèrent donc correctement le cas de plusieurs applications sur le même site web. Avec les autres navigateurs, tout se passe correctement si toutes les applications utilisent des noms différents pour l'entrée de formulaire de type `text` utilisée pour l'identifiant de l'utilisateur ; sinon on rencontre les mêmes problèmes d'automatisation et d'ergonomie que pour l'authentification HTTP Basic sans mémorisation du « realm ».

Gestion unifiée d'un mot de passe commun à plusieurs applications. Aucun des navigateurs testés ne dispose d'une fonctionnalité permettant de gérer en une seule opération un mot de passe commun à plusieurs applications web. L'utilisateur doit donc répéter une opération de gestion autant de fois qu'il y a d'applications.

Visualisation des mots de passe mémorisés. À l'exception d'Internet Explorer sous Mac OS X, tous les navigateurs permettent de visualiser les mots de passe :
– soit directement dans l'application pour Mozilla et Firefox ;
– soit à l'aide d'un outil externe fourni avec le navigateur pour Safari et Konqueror ;
– soit à l'aide d'un outil tiers pour Internet Explorer sous Windows (par exemple PStorageLib [4]) et Netscape (avec le bookmarklet `view password` [5]).

4.3 Sécurité des données mémorisées sur le poste client

Stockage des mots de passe. Tous les produits testés stockent les mots de passe mémorisés sous une forme codée, soit en chiffrant avec un algorithme de chiffrement réputé, soit en utilisant une transformation non documentée.

Lorsque l'utilisation d'un « master password » est possible, celui-ci est utilisé pour créer une clé de chiffrement.

Le tableau suivant résume la méthode de stockage utilisée selon application :

	Type de stockage	Algorithme de chiffrement	« Master password »
Mozilla, Netscape et Firefox	fichier dans le « homedir » de l'utilisateur	3DES	optionnel[a]
Internet Explorer sous Windows	base de registre	non documenté	non disponible
Internet Explorer sous Mac OS X	fichier dans le « homedir » de l'utilisateur	non documenté	non disponible
Safari [b]	fichier dans le « homedir » de l'utilisateur	3DES	obligatoire[c]
Konqueror [d]	fichier dans le « homedir » de l'utilisateur	Blowfish	obligatoire

[a] Lorsqu'il n'y pas de « master password », les mots de passe sont plus ou moins bien masqués (base64 ou 3DES avec une clé stockée en clair selon la configuration du navigateur).
[b] Stockage implémenté par le « trousseau de clé » de Mac OS X.
[c] Par défaut, le « master password » est initialisé avec le mot de passe de session Mac OS X de l'utilisateur.
[d] Stockage implémenté par l'application KDEWallet.

Grâce à divers tests, nous avons pu constater que les naviga-

teurs qui ne permettent pas d'utiliser de « master password » (Internet Explorer sous Windows et sous Mac OS X) ne se basent pas non plus sur le mot de passe de session du système pour protéger le stockage.

En effet, sous Mac OS X, une personne pouvant lire le fichier de stockage des mots de passe d'Internet Explorer[3] peut copier celui-ci dans son « homedir » et réutiliser les mots de passe mémorisés.

Sous Windows, les mots de passe sont stockés dans un « emplacement protégé » fourni par le système et géré par le service « Protected Storage ». Ce service de stockage « sécurisé » n'est pas documenté par Microsoft mais nous avons pu constater que le chiffrement est indépendant du mot de passe de session Windows de l'utilisateur et se base sur le SID de l'utilisateur. L'annexe « Accès aux mots de passe d'Internet Explorer sous Windows sans connaissance du mot de passe de session » décrit la méthode utilisée pour arriver à cette constatation.

Par conséquent, la protection fournie par le « Protected Storage Service » peut être contournée par une personne accédant au contenu de la base de registre (qui contient les mots de passe mémorisés d'Internet Explorer et le SID de l'utilisateur).

Ces deux implémentations de stockage ne sont donc pas suffisamment robuste en cas d'accès physique.

Durée de vie du « master password ». Pour limiter la présence du « master password » dans la mémoire du poste client, sa durée de vie est paramétrable. Les possibilités de configuration sont :
- toute la session du navigateur[4] (valeur par défaut pour tous les navigateurs) ;
- pour l'utilisation d'un seul mot de passe mémorisé (sauf Safari et Konqueror) ;
- pour une durée fixe configurable (sauf Konqueror) ;
- après une durée fixe configurable d'inutilisation des mots de passe stockés (seulement Konqueror) ;
- jusqu'au démarrage de l'économiseur d'écran (Safari et Konqueror) ;

Stockage des cookies. Aucun des navigateurs testés ne chiffre le stockage des cookies utilisés pour le suivi de session authentifiée par formulaire.

On peut noter que Mozilla, Netscape et Firefox permettent de forcer l'expiration d'un cookie à la fermeture du navigateur et ainsi éviter son stockage sur le disque dur.

Solutions de chiffrement externes. Tous les produits testés, à l'exception d'Internet Explorer sous Windows, utilisent un fichier comme moyen de stockage. Il est donc possible d'utiliser un outil de chiffrement de fichier comme alternative ou complément à l'utilisation d'un « master password ». Pour des considérations d'ergonomie, il faut que l'outil de chiffrement permette un accès transparent au fichier : c'est le cas, par exemple, de EFS sous Windows, CryptoAPI sous Linux et FileVault sous Mac OS X.

Pour Internet Explorer sous Windows, aucun outil ne permet de chiffrer la base de registre de manière transparente.

On peut noter que les solutions de chiffrement externes peuvent aussi être utilisées pour protéger le stockage des cookies.

Sauvegarde et restauration. Pour tous les produits se basant sur un fichier comme moyen de stockage, il n'y a pas de problème particulier pour la sauvegarde et la restauration.

Avec Internet Explorer sous Windows, il faut utiliser un outil permettant la sauvegarde de la base de registre.

4.4 Sécurité des échanges

Limitation de la mémorisation aux seules pages HTTPS. Aucun des navigateurs testés ne peut être configuré pour proposer la mémorisation des données d'authentification seulement lors d'accès aux pages HTTPS.

Réutilisation des mots de passe mémorisés en HTTPS. Dans le cas de l'authentification applicative par formulaire, Safari n'effectue aucune distinction entre les mots de passe mémorisés en HTTP et HTTPS lors de leur réutilisation. Les mots de passe mémorisés en HTTPS sont donc réutilisés pour s'authentifier sur le même site en HTTP. Tous les autres navigateurs testés réutilisent les données mémorisées en HTTPS seulement lors de l'accès au site légitime en HTTPS.

Dans le cas de l'authentification HTTP Basic, Safari et Konqueror réutilisent les données mémorisées en HTTPS seulement lors de l'accès au site légitime en HTTPS. Pour tous les autres produits testés, la mémorisation des mots de passe ne prend pas en compte le protocole utilisé (HTTP ou HTTPS) mais seulement le numéro de port TCP (par défaut, 80 pour HTTP et 443 pour HTTPS). Un mot de passe mémorisé pour l'accès à une page en HTTPS sur TCP 443 est donc réutilisé lors d'accès à cette même page sur le port TCP 443 que ça soit en HTTP ou HTTPS.

Ce comportement pourrait permettre à un attaquant de voler les authentifiants en forçant le navigateur de l'utilisateur cible à interroger, en HTTP sur TCP 443, une page du serveur cible dont le mot de passe a été préalablement mémorisé en HTTPS (en utilisant une image invisible factice comme pour les « web bugs » [6] par exemple). Comme en HTTP le serveur n'est pas authentifié, l'attaquant pourrait ensuite se faire passer pour le serveur cible en détournant le trafic réseau et donc voler les authentifiants.

Le risque lié à cette vulnérabilité est minimisé par les éléments suivants :
- Il n'existe apparemment pas d'outil permettant une exploitation facile.
- L'attaquant doit pouvoir mettre en place un détournement

[3] Par défaut, seul root et le propriétaire peuvent lire ce fichier.
[4] Pour Safari, la session est comprise entre le login et le logout de l'utilisateur.

de trafic réseau.
- L'utilisateur doit interagir en confirmant l'envoi des authentifiants (sauf sous certaines conditions avec Internet Explorer).

5 Conclusion

Attention, les fonctionnalités des outils évoluant, les conclusions présentées ci-dessous sont liées aux versions de logiciels étudiées.

La fonctionnalité de mémorisation des mots de passe dans les navigateurs web modernes, bien que ne permettant pas une automatisation complète de l'authentification, offre à l'utilisateur l'économie de la ressaisie et/ou de la mémorisation des mots de passe.

Toutefois, quelques inconvénients apparaissent clairement dans les résultats des tests :
- Dans trois cas d'utilisation, l'automatisation de l'authentification est dégradée voir impossible :
  - lors de la présence d'une liste déroulante dans un formulaire d'authentification ;
  - lorsque plusieurs applications web indépendantes fonctionnent sur le même site web ;
  - lorsque l'application web demande au navigateur de ne pas mémoriser le mot de passe en utilisant l'attribut `autocomplete`.
- L'ergonomie n'est pas satisfaisante :
  - lors d'un changement de mot de passe ;
  - lors de la gestion d'un mot de passe commun à plusieurs applications ;
  - lorsque l'on travaille depuis un poste qui n'a pas accès au système de fichiers habituel ;
  - lorsque l'on veut utiliser, provisoirement ou définitivement, un autre navigateur web.
- La sécurité ne peut pas être garantie :
  - en cas d'accès physique, la sécurité du stockage n'est pas assurée avec Internet Explorer sous Windows et Mac OS X, et ne le sera avec Mozilla, Netscape et Firefox qu'après la création d'un « master password » optionnel ;
  - il existe un risque de vol de mot de passe mémorisé en HTTPS par attaque réseau avec Mozilla, Netscape, Firefox, Internet Explorer sous Windows et Safari. Ce risque est plus élevé avec Internet Explorer car, sous certaines conditions, il peut intervenir sans interaction de l'utilisateur.

Malgré ces inconvénients, la mémorisation des mots de passe apporte un confort supplémentaire aux utilisateurs et peut être activée sans introduire de grosses failles de sécurité à condition de respecter quelques consignes :
- en cas d'utilisation d'Internet Explorer, n'activer la mémorisation que sur des postes dont l'accès physique est maîtrisé ;
- protéger le stockage à l'aide d'un « master password » robuste ;
- sauvegarder les fichiers de stockage des authentifiants.

5.1 Contexte d'applications web utilisant une base de comptes commune

Ce contexte correspond à la situation actuelle à l'INRIA où un nombre croissant d'applications web s'interface avec un service d'authentification basé sur un annuaire LDAP central ; ce qui permet à chaque utilisateur de disposer d'un mot de passe unique.

Malgré l'utilisation d'un mot de passe unique, l'utilisateur est amené à s'authentifier sur chaque serveur. Cette situation peut alors être perçue de manière négative par l'utilisateur ; en particulier, lorsque dans le cadre de l'ouverture d'une ressource pour les nomades, une authentification est nécessaire lors d'un accès à partir d'Internet (à la place d'un filtrage par adresse IP dans le réseau interne).

Pour éviter cette perception négative, des solutions de « Single Sign On » peuvent être mises en place. Une alternative, reposant sur la mémorisation des mots de passe par les navigateurs est parfois envisagée comme « Web Single Sign On ».

Si du point de vue utilisateur on aboutit à une situation acceptable il faut cependant garder en mémoire que :
- Du point de vue de l'ergonomie, aucun navigateur ne permet de regrouper la gestion de plusieurs mots de passe. Le changement du mot de passe est donc fastidieux.
- Si le mot de passe est compromis pour l'une des applications web, il peut être réutilisé pour s'authentifier sur toutes les applications. Pour minimiser le risque, la mémorisation du mot de passe ne doit être utilisée que si l'accès est en HTTPS (tout en prenant garde au risque de vol du mot de passe par attaque réseau pour certains navigateurs).

5.2 Contexte d'un ensemble hétérogène d'applications web

Ce contexte correspond à la situation que peut rencontrer un utilisateur qui crée des comptes, parfois temporaires, sur des sites web n'ayant aucun rapport entre eux (webmail, forum, inscription à une conférence, ...).

La mémorisation des données d'authentification par le navigateur est assez bien adaptée à ce type d'utilisation en particulier si l'utilisateur choisit des mots de passe différents (idéalement générés automatiquement et aléatoirement) pour chaque site afin que les conséquences de la compromission d'un mot de passe soient limitées.

Notons aussi qu'il existe une solution alternative (que nous n'avons pas testée en pratique) à la mémorisation des mots de passe pour ce contexte d'utilisation. Cette solution repose sur la création d'un mot de passe unique par site web à partir du hachage d'un « master password » et d'informations publiques du site web. La méthode est décrite en détail dans le document [7].

Annexe

Recommandations par application

Mozilla 1.7.3, Netscape 7.1 et Firefox 1.0. Malgré un risque faible de vol d'authentifiants par attaque réseau, l'usage des trois applications nous semble parfaitement envisageable à condition :
- de protéger le stockage à l'aide d'un « master password » robuste ;
- de sauvegarder les fichiers de stockage des authentifiants ;
- que l'utilisateur soit vigilant concernant l'identité du serveur lorsque qu'apparaît un pop-up prérempli.

Internet Explorer 6.0 sous Windows XP SP2. L'usage de la fonctionnalité de mémorisation d'authentifiants sensibles nous semble à proscrire car :

- Le stockage n'est pas sûr en cas d'accès physique à l'ordinateur.
- Un risque modéré de vol d'authentifiants par attaque réseau existe.

Si l'utilisateur choisit tout de même d'activer la mémorisation des mots de passe, il devra veiller à :
- ce que l'accès physique au poste de travail soit maîtrisé ;
- ce que son outil de sauvegarde permette de sauvegarder le contenu de la base de registre ;
- utiliser un compte de domaine (qui possède un SID unique utilisé sur tous les ordinateurs du domaine) afin que les authentifiants mémorisés puissent être réutilisés en cas de changement d'ordinateur.

Internet Explorer 5.2.3 sous Mac OS X. Nous recommandons de ne pas utiliser ce produit pour les raisons suivantes :
- Application trop ancienne, plus maintenue par Microsoft.
- La mémorisation de l'authentification par formulaire n'est pas disponible.
- Le stockage des données n'est pas sûr.

Safari 1.2.4 sous Mac OS X 10.3. Malgré un risque faible de vol d'authentifiants par attaque réseau, l'usage de cette application nous semble parfaitement envisageable à condition :
- d'utiliser un mot de passe de session Mac OS X robuste ;
- de sauvegarder les fichiers de stockage des authentifiants ;
- que l'utilisateur vérifie bien l'utilisation de HTTPS avant de valider un formulaire prérempli avec un mot de passe mémorisé en HTTPS.

Konqueror 3.4.2. L'usage de l'application nous semble parfaitement envisageable à condition :
- de protéger le stockage à l'aide d'un « master password » robuste ;
- de sauvegarder les fichiers de stockage des authentifiants.

Accès aux mots de passe d'Internet Explorer sous Windows sans connaissance du mot de passe de session

Cette annexe décrit une méthode utilisable pour accéder aux mots de passe d'Internet Explorer sous Windows XP SP2 sans connaître le mot de passe de session de l'utilisateur cible. Elle n'est pas optimale et a pour seul but de démontrer que la visualisation des mots de passe est possible en cas d'accès physique.

Il faut d'abord déterminer si le compte cible est local ou du domaine :

1. Se connecter en tant qu'administrateur local sur l'ordinateur cible. L'outil ntpasswd [8] peut être utilisé pour réinitialiser le mot de passe.
2. Récupérer le SID de l'ordinateur avec PsGetSid de Sysinternals [9] en utilisant la commande `psgetsid \\nom_machine_cible`
3. Récupérer le SID de l'utilisateur cible dans la base de registre en recherchant son nom de login dans une clé se trouvant sous `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\ProfileList` (le SID recherché est dans le chemin contenant cette clé).

Si les 7 premiers champs du SID de l'utilisateur correspondent au SID de l'ordinateur, c'est un compte local ; sinon, c'est un compte de domaine.

Cas d'un compte local. Dans ce cas, l'accès aux mots de passe est simple et ne nécessite que très peu d'opérations :

1. Modifier le mot de passe du compte local cible (en utilisant l'utilitaire « Gestion de l'ordinateur »).
2. Se connecter sous le compte local cible avec le nouveau mot de passe.
3. Utiliser PStorageLib [4] pour visualiser les mots de passe.

Cas d'un compte de domaine. Ce cas est plus compliqué car il ne semble pas exister de moyen simple de modification du mot de passe de l'utilisateur cible sans connaître l'ancien mot de passe ou être administrateur du domaine. La méthode utilisée consiste donc à créer, sur un deuxième ordinateur, un utilisateur avec un mot de passe connu et possédant le SID de l'utilisateur cible.

Voici les détails de la procédure :

1. Sur un deuxième ordinateur (utilisant aussi Windows XP SP2) en tant qu'administrateur local, créer un compte local avec le 8e champ du SID correspondant au 8e champ du SID de l'utilisateur cible. Nous avons écrit le script `creationUtilisateurSID.vbs` [10] dans le but d'automatiser cette tâche.

   Par défaut, le compte créé se nomme `comptelocal` et a pour mot de passe `comptelocal`.
2. Se déconnecter puis se connecter sous `comptelocal` (afin de créer le répertoire de profil de l'utilisateur).

3. Se déconnecter puis se connecter en tant qu'administrateur local.
4. Copier le fichier caché `NTUSER.DAT` à partir du répertoire de profil de l'utilisateur cible sur l'ordinateur cible (sous l'identité administrateur local) vers le répertoire de profil de `comptelocal` sur le deuxième ordinateur (sous l'identité administrateur local). Plusieurs méthodes sont possibles pour le transfert : clé USB, CD gravé, montage SMB, ...
5. Modifier le SID du deuxième ordinateur en utilisant l'utilitaire NewSID de Sysinternals [11]. Il faut fournir les 7 premiers champs du SID de l'utilisateur cible.
6. Redémarrer le deuxième ordinateur.
7. Se connecter sous `comptelocal`.
8. Utiliser PStorageLib [4] pour visualiser les mots de passe.

Références

[1] Rfc 2617 : Http authentication : Basic and digest access authentication. `http://www.ietf.org/rfc/rfc2617.txt`.

[2] Rfc 2965 : Http state management mechanism. `http://www.ietf.org/rfc/rfc2965.txt`.

[3] Description de l'attribut autocomplete. `http://whatwg.org/specs/web-forms/current-work/#the-autocomplete`.

[4] Site web du logiciel pstoragelib. `http://www.cobans.net/pslib.php`.

[5] Bookmarlets concernant les formulaires. `http://www.squarefree.com/bookmarklets/forms.html`.

[6] Faq sur les « web bugs ». `http://www.eff.org/Privacy/Marketing/web_bug.html`.

[7] Blake Ross et Collin Jackson et Nicholas Miyake et Dan Boneh et John C. Mitchell. Stronger password authentication using browser extensions. Dans *Proceedings of the 14th Usenix Security Symposium*, 2005. `http://crypto.stanford.edu/PwdHash/pwdhash.pdf`.

[8] Site web du logiciel ntpasswd. `http://home.eunet.no/~pnordahl/ntpasswd/`.

[9] Site web du logiciel psgetsid. `http://www.sysinternals.com/ntw2k/freeware/psgetsid.shtml`.

[10] Page web contenant le script `creationutilisateursid.vbs`. `http://hal.inria.fr/inria-00000404`.

[11] Site web du logiciel newsid. `http://www.sysinternals.com/Utilities/NewSid.html`.