

Sécurité logicielle: Protection logicielle, buffers overflows

Cours de sécurité: Licence Pro Alternance

9 Juillet 2008

1 Protection logicielle

1.1 Exercice 1

Un programmeur a décidé de protéger son application à l'aide d'une validation de type license. N'ayant pas suivi de cours de sécurité, il a décidé de protéger son application lui-même. Son application est disponible à l'adresse www.enseirb.fr/~tabary/secu/tp3/exo1.

1. A l'aide de l'outil OllyDbg, retrouvez le numéro de license en consultant simplement le code assembleur de l'application
2. Que peut-on dire de ce type de vérification ?

1.2 Exercice 2

Notre programmeur ne se décourage pas. Il a compris que laisser la clé de license en clair dans le code réduit la sécurité de son application. Il a donc décidé de la protéger d'une autre manière. Son application est disponible à l'adresse www.enseirb.fr/~tabary/secu/tp3/exo2.

1. Modifiez l'application pour que son programme ne vérifie plus la clé de license
 - L'instruction `nop`, qui n'effectue aucune opération, pourra vous aider
2. Comment feriez-vous pour retrouver la clé ?
3. Que penser de cette protection ? Comment l'améliorer ?

1.3 Exercice 3

Ne voulant pas admettre ses torts, notre programmeur met à jour son application avec une protection plus subtile. Son application est disponible à l'adresse www.enseirb.fr/~tabary/secu/tp3/exo3.

1. Récupérez la clé de license
 - Note: il va falloir débogger cette fois-ci
2. Quel principe de sécurité a-t-il mis en place ?
3. Que penser de cette protection ?

1.4 Exercice 4

Notre programmeur a lu sur internet quelques conseils pour empêcher le piratage de son application. Ne s'avouant pas vaincu, il met en place une dernière protection. Son application est disponible à l'adresse www.enseirb.fr/~tabary/secu/tp3/exo4.

1. Cassez la protection
2. Comment aurait-il fallu intégrer à l'application l'astuce du programmeur pour qu'elle soit plus efficace ?

1.5 Conclusion

Le piratage d'applications est souvent dû à des programmeurs pensant qu'ils sont capables de protéger eux-mêmes leurs logiciels. C'est un erreur qui profite aux pirates et qui nuit aux entreprises. Utiliser des protections éprouvées et sûres est nécessaire pour garantir le respect de la propriété intellectuelle.

2 Failles logicielles

2.1 Exercice 5

Un service réseau tourne sur une machine sur le port 5678 (adresse IP au tableau). Ce programme nécessite un mot de passe avant de pouvoir être utilisé. Connaissant le programmeur à l'origine de l'application, vous suspectez que le programme est vulnérable à une attaque de type **stack overflow**.

1. Exploitez le programme pour pouvoir l'utiliser même sans connaître le mot de passe
 - (a) Quelle est la taille du buffer d'entrée pour la demande de mot de passe ?
 - (b) Construisez une chaîne qui fera apparaître un stack overflow (on pourra la construire dans un fichier à l'aide de l'éditeur hexadécimal **hiew** avant de l'envoyer au programme)
 - (c) Modifiez la chaîne pour que la fonction `autoriser_acces` soit exécutée
2. A votre avis, quelle fonction a utilisée le programmeur pour lire le mot de passe ?
3. Quelles protections permettraient d'éviter cette faille logicielle ?

2.2 Exercice 6

L'erreur du programmeur peut avoir des conséquences bien plus lourdes qu'un accès non autorisé à l'application. Une faille de type **remote shell** est en effet possible.

1. Exploitez le programme présent sur le port 5679
 - Le buffer est de 460 octets cette fois-ci
 - Aidez-vous du fichier `shellcode.txt` présent dans le répertoire `tp3/exo6`
2. Quelle protection aurait pu déjouer cette attaque ?

2.3 Conclusion

Les failles logicielles sont une réalité qui coute très cher aux entreprises chaque année. Une simple erreur de programmation peut avoir des conséquence désastreuses. Pensez-y lorsque vous devrez développer au sein d'une entreprise !