ZIMPERIUM®

10 façons de protéger les applications

Les protections in-app modifient un l'application dans le but de la rendre plus résistante à la rétro-ingénierie, à la falsification ou encore à la surveillance.

Vous trouverez ci-dessous les dix techniques les plus recommandées, depuis l'obfuscation la plus basique jusqu'à l'équipement de votre application avec un système autonome de détection et de réaction face aux attaques.

1 Modifier noms

Remplacer les noms de variables et de méthodes identifiables par des chaînes de caractères sans signification. Ils deviendront donc illisibles et difficiles à comprendre par un pirate. Il existe plusieurs outils d'obfuscation gratuits qui servent à attribuer de nouveaux noms aux classes, aux champs ainsi qu'aux méthodes.

2 Insérer Code factice

Permet d'ajouter un code non fonctionnel dans l'application dans le but de complexifier l'analyse du code de rétro-ingénierie. Il existe plusieurs obfuscateurs gratuits qui insèrent, à leur tour, un code factice.

3 Retirer le code inutile ainsi que les métadonnées

Éliminer le code mort, les informations de débogage et les métadonnées non essentielles des applications dans le but de limiter les informations qu'un attaquant peut collecter et exploiter.

4 Appels à partir du code Objective C.

Les messages appelés en Objective-C sont résolus pendant l'exécution, ils sont donc stockés en clair dans le code, dans lequel ils peuvent éventuellement être exploités. Protéger le code Objective-C en obfusquant les appels de messages dont le texte est clair pour qu'ils soient difficiles à lire et à modifier.

5 Utiliser le package binaire

Ajoutez votre code à un paquet pour le protéger contre l'analyse statique. Les packers permettent de chiffrer et de compresser le code, en y ajoutant un bouchon qui le décompose pendant l'exécution. Ainsi, les pirates n'arriveront pas à réaliser de l'ingénierie inverse, puisqu'ils ne pourront en aucun cas exécuter le code par un désassembleur ou un décompilateur.

6 Protection contre les débogueurs

Les pirates utilisent habituellement des débogueurs dans le but de comprendre le fonctionnement d'une application et de réaliser une ingénierie inverse. Pour un antidébogage basique, insérez des appels API pour collecter des informations sur le processus et le système dans le but de détecter la présence d'un débogueur. Pour une protection optimale du débogueur, optez pour un outil capable de détecter les points d'arrêt insérés dans le débogueur et d'exécuter automatiquement des actions défensives.

7 Diversifiez vos logiciels

Créez plusieurs instances logicielles identiques sur le plan fonctionnel, mais dont le code se distingue de façon unique par sa forme et sa structure. Cette démarche obligera les éventuels pirates à décrypter chaque copie de l'application indépendamment, plutôt que de mener une seule attaque sur toutes les instances de l'application.

8 Obfusquer le flux de contrôle

Modifier la structure de base de la manière dont les sous-routines sont appelées pour semer confusion et rendre le code indéchiffrable. À titre d'exemple, les fonctions en ligne (inline) substituent les appels des sous-routines par des sauts calculés et aplatissent la structure de contrôle en convertissant les instructions conditionnelles ¬sous forme de branchement en instructions plates de type switch.

9 Protection contre l'altération

Mettez en place des protections in-app qui permettent d'éviter toute tentative de modifier ou détourner le code de l'application - insérez des sommes de contrôle superposées pour contrôler l'intégrité du code, ajoutez des réglages qui vous permettront de contourner la détection de jailbreak dans l'iOS et enracinez vos dispositifs Android, procédez au contrôle ¬croisé de vos bibliothèques partagées, évitez le swizzling de méthode, vérifiez l'identité de tout appelant de fonction et utilisez d'autres moyens de protection anti-sabotage.

10 Programmez votre application pour qu'elle se défende -automatiquement

Quand votre application détecte une tentative d'altération, elle doit déclencher une réponse défensive adéquate. Si possible, intégrez des mécanismes d'auto-protection en temps réel au lieu de générer des alarmes qui doivent être acquittées manuellement. Les mesures de protection habituelles comprennent le blocage de l'accès au compte, l'arrêt de l'exécution des commandes, la suppression des données sensibles et la fermeture totale de l'application. Dans le but d'éviter d'éventuelles tentatives d'attaque, essayez de corrompre certaines parties de l'application, ainsi, le pirate va croire qu'il a réussi, alors qu'il ne dispose en fait que d'un accès très limité.

Zimperium aide les organisations à créer des applications mobiles sécurisées et conformes. Il s'agit de l'unique solution unifiée qui combine une protection in-app complète avec une visibilité centralisée des menaces. Cliquez ici pour en savoir davantage.

ZIMPERIUM®

4055 Valley View Dallas, TX 75244
844.601.6760
info@zimperium.com