ZIMPERIUM®

Pourquoi vous devez protéger vos clés cryptographiques

Contents

Résumé analytique

Au cours des dernières décennies, la sécurité des données était déterminée par des ressources informatiques massives contrôlées par des entreprises et des centres de données physiquement sécurisés. La réalité actuelle est que de nombreuses applications logicielles s'exécutent sur des dispositifs non gérés dans des réseaux vulnérables et ciblés. Des personnes mal intentionnées peuvent facilement accéder physiquement à de nombreux dispositifs destinés à protéger des secrets internes, y compris des téléphones mobiles, des appareils de l'IoT, des automobiles, des décodeurs et des équipements médicaux. Même dans un environnement d'entreprise bien sécurisé, le périmètre est de plus en plus difficile à définir et à défendre, car certains dispositifs accédant au réseau ne sont pas correctement gérés ou sécurisés (par exemple, ceux de type BYOD, à savoir la pratique qui consiste à utiliser ses équipements personnels dans un contexte professionnel). Les déploiements de programmes malveillants à grande échelle font qu'il est probable que des dispositifs soient infectés, et ce quelle que soit la qualité de leur gestion. Par conséquent, il existe un risque élevé que des personnes mal intentionnées puissent facilement examiner et attaquer ces types de dispositifs.

Dans ce document, nous nous concentrerons sur un risque particulier pour la sécurité qui est inévitable dans les environnements numériques ouverts et non sécurisés d'aujourd'hui, à savoir la sécurité des clés cryptographiques. Comme nous l'expliquerons plus en détail, une clé cryptographique est le concept fondamental de la plupart des systèmes de sécurité utilisés sur des milliards de dispositifs dans le monde entier. Bien que la cryptographie soit conçue pour assurer la protection des données confidentielles, elle n'élimine pas automatiquement le risque qu'elles soient attaquées, car la sécurité cryptographique repose sur la sécurisation des clés. En réalité, la cryptographie ne fait que déplacer le problème de la protection des données vers celle des clés.

En fait, dans de nombreuses situations, une seule clé peut protéger de nombreux éléments de données différents, la sécurisation de telles clés est donc d'une importance capitale. L'extraction injustifiée d'une clé d'un module cryptographique revient essentiellement à désactiver l'ensemble du système de sécurité. Les conséquences d'une clé compromise peuvent inclure une perte financière, un recours en responsabilité, des pénalités réglementaires et un impact sur la réputation de la marque.

Nous présenterons une vue d'ensemble des techniques fréquemment utilisées par les pirates pour découvrir les clés, telles que l'utilisation de l'analyse statique et dynamique, l'écoute clandestine du réseau et les attaques sur les canaux latéraux. En outre, les méthodes établies pour lutter contre ces attaques seront également discutées, et le concept de cryptographie en boîte blanche sera expliqué.

Enfin, ce livre blanc se concentrera sur la solution leader de l'industrie d'Intertrust pour la protection des clés cryptographiques dans les logiciels — **zKeyBox de Zimperium**, une bibliothèque de cryptographie en boîte blanche fournissant une implémentation sécurisée des algorithmes cryptographiques standard qui **masque complètement les clés cryptographiques** dans le code binaire et rend les tentatives d'extraction de clés extrêmement difficiles.

Le défi posé par la sécurité des clés cryptographiques

La **cryptographie** est la base de la sécurité des données dans les ressources et services numériques utilisés par des millions de personnes chaque jour. Elle permet une communication sécurisée, une authentification solide et la protection des informations confidentielles. Cartes bancaires, DAB, télévision payante, cloud computing, paiements en ligne et voitures connectées ne sont que quelques exemples de systèmes modernes qui seraient très vulnérables et peu pratiques sans l'utilisation de la cryptographie.

Au cœur de la cryptographie se trouve le concept d'une clé — un petit élément d'information qui détermine le résultat des opérations cryptographiques (cryptage, décryptage, signature, vérification, etc.). L'accès à la bonne clé donne accès à toutes les données secrètes protégées par l'algorithme cryptographique s'y rapportant. Bien que ceux qui utilisent des algorithmes cryptographiques admettent généralement la nécessité de protéger leurs données secrètes, le besoin de sécuriser les clés cryptographiques elles-mêmes est souvent négligé. Une hypothèse fallacieuse est que les clés cryptographiques secrètes ne sont pas accessibles aux personnes mal intentionnées. Dans la grande majorité des cas, les algorithmes cryptographiques exposent les clés qu'ils chiffrent d'une manière ou d'une autre dans l'environnement d'exécution. Il existe de nombreuses façons d'obtenir les clés, comme expliqué plus loin dans la section « Moyens permettant aux personnes mal intentionnées de s'attaquer aux clés cryptographiques ». Par conséquent, l'un des aspects principaux à souligner est qu'il **est absolument essentiel de protéger les clés cryptographiques.**

Si des pirates informatiques parviennent à obtenir des clés cryptographiques, ils peuvent éventuellement espionner une communication sécurisée, usurper un utilisateur, manipuler des transactions réseau et/ou infiltrer le système pour en extirper des informations confidentielles. Les conséquences de modules cryptographiques rendus inopérants et de clés volées peuvent être importants pour les gouvernements, les institutions financières, les constructeurs automobiles, les organismes de santé et les distributeurs de jeux. Des pertes financières, une atteinte à la réputation de la marque, une exposition à un recours en responsabilité et parfois même un décès peuvent tous résulter de l'incapacité à assurer une protection adéquate des clés cryptographiques.

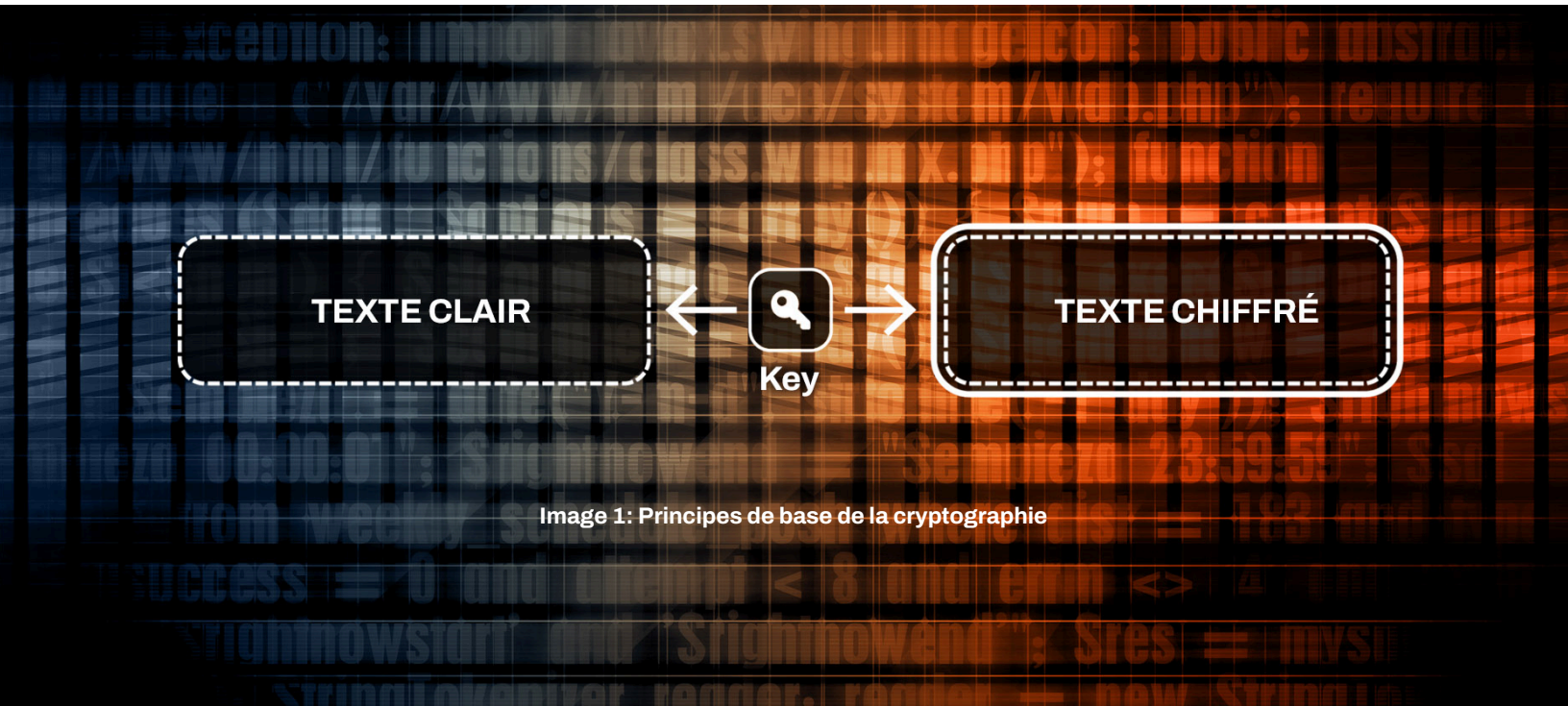


Image 1: Principes de base de la cryptographie

Exemples d'attaques visant des clés cryptographiques

Voici quelques-unes des attaques bien connues visant de grandes organisations et impliquant la révélation de clés cryptographiques.

Télécommandes Volkswagen

En 2016, une équipe d'informaticiens a publié un article sur un défaut dont sont victimes pratiquement toutes les voitures que Volkswagen a vendues depuis 1995. En utilisant un matériel radio bon marché et facilement disponible, ils ont pu intercepter les signaux de la télécommande d'une victime, découvrir les clés secrètes utilisées, puis cloner la télécommande d'origine[1].

Application mobile Tesla

En 2016, une équipe d'experts en sécurité a fait état d'une vulnérabilité qui leur a permis de prendre le contrôle intégral d'un modèle S de Tesla en déjouant les mesures de sécurité de l'application mobile Tesla. L'application est authentifiée à l'aide d'une clé secrète qu'elle stocke localement. Étant donné que la clé était stockée sans chiffrement, elle pouvait être dérobée par des programmes malveillants placés sur un appareil mobile. Cette vulnérabilité a été exploitée dans la pratique en installant une version malveillante de l'application mobile Tesla.

Console Nintendo Wii

En 2007, un pirate a pu obtenir des clés de chiffrement secrètes utilisées sur la console Nintendo Wii en exploitant un bogue dans l'algorithme de vérification de signature et en compromettant les clés qui étaient stockées dans la mémoire RAM GDDR3 externe sous forme non chiffrée. En conséquence, les mécanismes antipiratage de la console ont été inefficaces, permettant l'installation et l'exécution de logiciels non approuvés sur la Wii[2].

Moyens qu'utilisent les pirates pour s'attaquer aux clés cryptographiques

Cette section présente les méthodes courantes utilisées par les pirates pour extraire des clés secrètes de divers systèmes. Elle souligne que l'extraction de clés représente un risque sérieux et que la protection de vos systèmes contre de telles attaques devrait être une priorité.

Attaque par force brute

Lors d'une attaque par force brute, le pirate tente un grand nombre d'entrées pour voir si l'une d'elles fonctionne. Par exemple, de nombreux algorithmes de piratage de mot de passe (Brutus, RainbowCrack) fonctionnent de cette façon, en essayant des millions de mots de passe communs jusqu'à trouver celui qui fonctionne. C'est pourquoi il vous est toujours demandé de choisir des mots de passe avec des combinaisons difficiles à mémoriser de lettres en majuscules et en minuscules, de chiffres et de caractères spéciaux.

En général, les attaques par force brute ne sont efficaces que pour briser les algorithmes cryptographiques qui traitent de petites tailles de clés. Avec les derniers algorithmes cryptographiques standard du secteur, les attaques par force brute sont généralement impossibles.

Vulnérabilités théoriques et erreurs d'implémentation

Des pirates peuvent tenter de détecter des faiblesses théoriques ou des bogues de mise en œuvre dans les algorithmes ou protocoles cryptographiques qui leur permettraient de contourner rapidement les protections de sécurité inhérentes à un algorithme ou un protocole particulier. Un exemple classique de ceci est l'attaque « man-in-the-middle » contre le protocole à clé publique Needham-Schroeder[3]. Cette attaque a révélé une faiblesse fondamentale dans le protocole qui a permis à une attaque imprévue de réussir. Le protocole WEP (Wired Equivalent Privacy) est un autre cas où une vulnérabilité théorique a été découverte et publiée dans une publication[4]. Un exemple plus récent qui a tiré parti d'une faille dans la bibliothèque de logiciels cryptographiques OpenSSL[5] fut la fameuse vulnérabilité Heartbleed.

Comme l'ont démontré ces exemples, même des normes et des systèmes bien établis risquent d'être attaqués et compromis.

Analyse statique

En analysant le code machine statique d'un exécutable logiciel tel que l'image binaire dans le stockage du dispositif, les pirates peuvent facilement découvrir les clés cryptographiques si elles sont stockées sans protection. L'identification de clés potentielles dans le code est facilitée par le fait que les clés cryptographiques sont des ensembles aléatoires de bits présentant une entropie élevée. En revanche, la plupart des codes machine non compressés ont une entropie relativement faible6. Par conséquent, une clé est susceptible de se démarquer en arrière-plan de données à faible entropie ne s'y rapportant pas. L'image suivante montre le code machine en 2D, de sorte qu'un pixel représente un bit, et chaque colonne représente 64 bits de données séquentielles (de gauche à droite). L'œil humain peut rapidement identifier une région caractérisée par un caractère aléatoire élevé, ce qui peut indiquer une clé cryptographique. Le processus de localisation de telles zones peut être facilement automatisé.

L'analyse statique est l'une des attaques les plus efficaces si le pirate a accès au stockage du dispositif ou à tout canal utilisé pour déployer le code exécutable, et si les clés sont stockées sous forme de texte clair.

Image 2 : découverte de matériel clé à entropie élevée dans le code binaire

Analyse dynamique (ou de la mémoire)

While encrypting a key on a storage medium is a fairly simple procedure, hiding the key in device memory is much more complicated because at some point, the key needs to be provided to a cryptographic algorithm as valid input. In most cryptographic libraries this is the moment when the key is decrypted in the memory as plaintext and becomes susceptible to extraction. With the right set of tools, attackers can dynamically analyze the memory and hijack cryptographic secrets during execution of the software. There are automated tools which are readily available that can instantly discover secret keys in any arbitrary process running on a device[7].

Écoute clandestine sur une communication réseau

Secret keys should never be transferred over any network in unencrypted form, as this enables threat actors to easily exploit keys. From a security point of view, the Internet should be viewed as a completely transparent ecosystem where hackers can potentially see all the data you exchange with other endpoints. Consequently, it becomes absolutely clear that cryptographic keys and other secrets sent through the Web must always be protected. The common practice is to encrypt all secrets before they are sent over the Internet, and never expose these keys used for encrypting the secrets. There are established methods for agreeing on encryption keys on both endpoints without sending them over the Internet, such as the Diffie-Hellman key exchange algorithm[8].

Attaques sur les canaux latéraux

Dans ces attaques, le pirate ne tente pas d'accéder à la clé directement dans le dispositif, mais plutôt de la reconstruire à partir de signaux indirects et de la physiologie des composants internes du dispositif. Par exemple, dans certains cas, il est possible de reconstruire une clé en mesurant la consommation électrique d'une puce[9].

Dans un autre exemple, l'attaquant injecte des erreurs dans l'algorithme en soumettant le matériel à des températures extrêmes, puis observe le comportement de l'algorithme afin de reconstruire la clé[10].

Dans certaines circonstances, les clés peuvent être extraites des dispositifs même lorsqu'ils sont éteints. Ce type d'attaque sur les canaux latéraux repose sur la rétention de mémoire fréquente dans la plupart des dispositifs modernes. Même après la mise hors tension du dispositif, la mémoire interne conserve son contenu pendant quelques secondes à quelques minutes à des températures de fonctionnement normales, même si elle est retirée d'une carte mère[11]. Pour exécuter l'attaque, un redémarrage à froid du dispositif est effectué et un disque amovible est ensuite immédiatement utilisé pour lancer un système d'exploitation léger, ou dans certains cas, les modules de mémoire sont retirés du système d'origine et placés rapidement dans un ordinateur compatible. Une analyse plus poussée peut ensuite être effectuée à partir des informations qui ont été vidées de la mémoire pour trouver les clés cryptographiques qu'elles contiennent. Des outils automatisés sont désormais disponibles pour exécuter cette tâche en cas d'attaques contre certains systèmes de chiffrement courants.

Méthodes de protection des clés cryptographiques

Dans la section précédente, nous avons expliqué l'importance de cacher et de protéger les clés cryptographiques — un fait qui est souvent ignoré, même par les grandes sociétés. En même temps, nous avons montré qu'il n'est pas facile d'assurer une bonne protection des clés, car il existe un large éventail de techniques que les pirates utilisent pour attaquer les systèmes cryptographiques et les voler.

Dans cette section, nous décrivons les principales catégories de contre-mesures permettant d'éviter la révélation de clés cryptographiques.

Sécurité matérielle

Pour faire face aux défis posés par la protection, la sécurité matérielle est généralement utilisée pour fournir une sécurisation renforcée des clés sur les dispositifs. Par exemple, les modules matériels de sécurité (HSM), les modules de plateforme de confiance (TPM) et les environnements d'exécution de confiance (TEE). La sécurité de ces systèmes repose sur le fait qu'il est très difficile et coûteux pour les attaquants de procéder à l'ingénierie inverse d'un module matériel et de manipuler ses données internes. En règle générale, les systèmes matériels de sécurité peuvent être considérés comme des « modèles en boîte noire » parce que leur fonctionnement interne est essentiellement caché à l'observateur.

Bien que les approches basées sur le matériel offrent d'excellents avantages en matière de sécurité, elles entraînent également des inconvénients importants :

- La sécurité matérielle **renchérit les coûts** d'un système. Les fabricants de plates-formes peuvent être davantage sensibles aux coûts plutôt qu'aux risques pour la sécurité liés aux clés compromises. On réfléchit alors à la sécurité après coup.
- Les approches basées sur le matériel sont **difficiles et potentiellement coûteuses à atténuer**. Des exemples comme Meltdown et Spectre[12] illustrent que les fabricants de matériel et de logiciels peuvent avoir besoin de dépenser beaucoup d'argent et de ressources pour émettre des correctifs afin de corriger les vulnérabilités dans les déploiements existants.
- Différents dispositifs peuvent contenir **divers matériels avec des fonctionnalités variables** qui nécessitent une logique complexe dans les applications conçues pour fonctionner sur une large gamme d'entre eux.
- Comme il a été expliqué dans la section « Attaques sur les canaux latéraux », **le matériel n'est pas à l'abri des attaques**. Des approches intelligentes, telles que l'analyse de puissance différentielle, peuvent être utilisées pour extraire des clés du matériel en examinant des schémas indirects dans les signaux émanant du matériel.
- Il existe des modèles d'entreprise qui **empêchent les développeurs d'applications d'utiliser du matériel sécurisé** sur un dispositif, même s'il existe. Tel est le cas avec l'iPhone d'Apple qui, bien que doté de processeurs ARM avec l'extension TrustZone, n'autorise pas les applications tierces à utiliser cette fonctionnalité.

Magasins de clés (ou keystores)

La plupart des systèmes d'exploitation et des plates-formes d'exécution offrent des moyens de stockage et d'utilisation sécurisés des clés cryptographiques. Par exemple : Android Keystore, Java Keystore, Apple Secure Enclave et Windows Keystore. Dans certains cas, ces magasins de clés sont soutenus pas une sécurité matérielle, si cette technologie est disponible sur l'appareil. En général, les magasins de clés sont utilisés pour la gestion des certificats et des paires de clés associés à la communication SSL.

Bien que de tels magasins de clés soient suffisamment sécurisés, ils ne peuvent pas être considérés comme des bibliothèques cryptographiques à usage général. Par exemple, la liste des algorithmes et opérations cryptographiques pris en charge est généralement assez limitée. En outre, dans certains cas, il n'est pas possible d'importer une clé existante dans le magasin de clés. Un autre facteur important à prendre en compte est que de tels magasins de clés sont construits pour une plate-forme cible particulière, ce qui signifie que la prise en charge de la même application sur plusieurs plates-formes nécessitera de remettre en œuvre des opérations cryptographiques sur chacune d'elles. Pour ces raisons, il peut être difficile et coûteux de recourir à un magasin de clés spécifique à une plate-forme, selon le cas d'utilisation.

Cryptographie en boîte blanche

L'objectif de la cryptographie en boîte blanche est de mettre en œuvre des primitives cryptographiques de manière à ce que, dans le contexte de l'application prévue, **disposer d'un accès complet à l'implémentation cryptographique ne présente aucun avantage pour une personne mal intentionnée** par rapport à une situation où cette dernière travaillerait sur l'implémentation sous forme de boîte noire[13]. En termes simples, la cryptographie en boîte blanche est une implémentation logicielle générale d'algorithmes cryptographiques qui vise à masquer des clés. Comme le logiciel peut être facilement examiné si le pirate a accès au dispositif, un tel environnement d'exécution logicielle est appelé un « modèle en boîte blanche ».

Travaux universitaires sur la cryptographie en boîte blanche

L'idée à la base de la cryptographie en boîte blanche peut sembler une formule magique impossible, mais les chercheurs universitaires étudient le problème de l'obfuscation générale depuis 2001. Au fil du temps, plusieurs dérivés universitaires de la cryptographie en boîte blanche ont émergé, comme les suivants :

- Le **chiffrement fonctionnel** (depuis 2005) combine le chiffrement de base avec un contrôle d'accès mathématiquement créé.
- Le **chiffrement entièrement** (depuis 2009) permet de sécuriser l'informatique avec des données chiffrées sur un serveur dans le cloud non approuvé.
- L'**obfuscation** (depuis 2013) permet d'obtenir une obfuscation logicielle générale qui a été appelée « crypto-complète » car un flux d'applications cryptographiques exotiques peut être créé à partir d'une obfuscation indiscernable.

Bien que la plupart de ces techniques cryptographiques avancées soient théoriquement possibles, elles sont pratiquement impossibles à appliquer, car elles nécessitent d'énormes quantités de ressources de calcul pour résoudre les problèmes les plus simples. Ce sont des domaines d'enquête actifs dans lesquels les chercheurs ne cessent de progresser. Toutefois, il faudra peut-être attendre des décennies avant que ces techniques ne soient mises en pratique.

Fonctionnement de la cryptographie en boîte blanche

Pour implémenter des primitives cryptographiques en boîte blanche, il est nécessaire de fournir des fonctionnalités équivalentes aux algorithmes standard sans révéler les valeurs intermédiaires au sein des algorithmes habituels. Une technique générale consiste à coder et donc à masquer les entrées, les sorties et les valeurs intermédiaires. Une autre technique consiste à réorganiser les étapes en opérations combinées moins révélatrices.

Comme l'illustre cette image, dans une implémentation « régulière » ou non brouillée, les clés secrètes et la logique d'exécution sont clairement distinguables et faciles à manipuler. Dans une implémentation en boîte blanche, les données internes et le flux d'exécution sont obscurs et inséparables — les clés ne peuvent pas être facilement extraites et apporter des modifications au code peut entraîner la rupture de l'exécutable dans son ensemble. L'une des méthodes les plus courantes dans les implémentations en boîte blanche consiste à déplacer les calculs dans des tables qui peuvent être facilement randomisées et à partir desquelles il est difficile d'effectuer une ingénierie inverse.

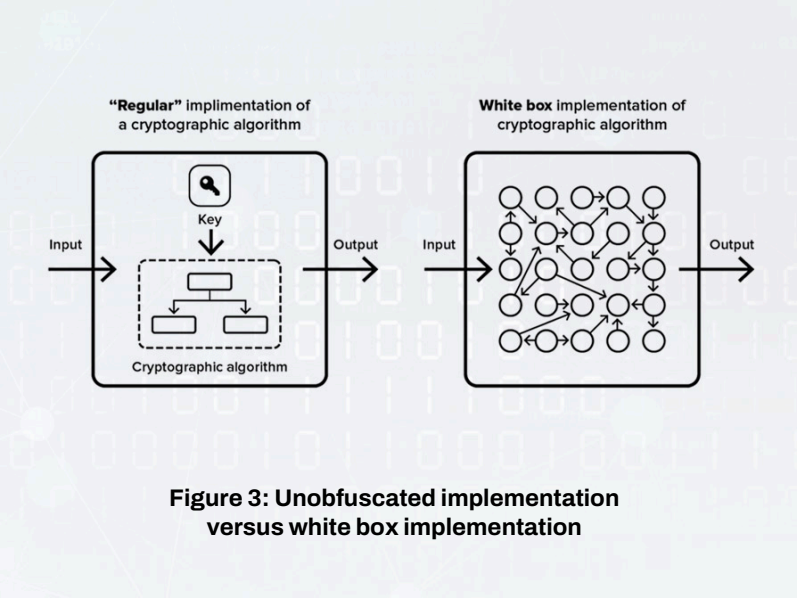


Figure 3: Unobfuscated implementation versus white box implementation

Choisir la bonne technique de protection des clés

En règle générale, la sécurité logicielle ne peut pas être considérée comme offrant la protection d'un matériel de sécurité dédié, et les calculs effectués dans un environnement logiciel en boîte blanche seront toujours plus lents. Cependant, l'avantage évident des algorithmes logiciels en boîte blanche par rapport à leurs homologues matériels en boîte noire est qu'ils peuvent être déployés sur des dispositifs sans prise en charge matérielle. Les algorithmes logiciels en boîte blanche peuvent prendre en charge les mêmes fonctionnalités sur n'importe quelle plate-forme et peuvent être mis à niveau facilement et de manière rentable si des vulnérabilités sont détectées. Dans certains cas, il peut être souhaitable de mettre en place une protection logicielle et matérielle pour assurer une défense en profondeur. Tous ces facteurs doivent être soigneusement évalués lors du choix de la technique souhaitée de protection des clés.

zKeyBox de Zimperium est la première implémentation au monde d'algorithmes de cryptographie en boîte blanche qui fournit une solution robuste au problème de la sécurisation des clés dans les logiciels et assure une protection contre la grande majorité des attaques visant des clés, y compris l'analyse statique et dynamique ainsi que les attaques sur les canaux latéraux.

La partie suivante de ce livre blanc portera sur la bibliothèque zKeyBox et sur la façon dont les fonctionnalités particulières traitent les différentes menaces visant les clés cryptographiques et d'autres parties internes des algorithmes cryptographiques.

zKeyBox de Zimperium

zKeyBox est une bibliothèque multi-plate-forme qui fournit une **implémentation en boîte blanche avancée d'un certain nombre d'algorithmes cryptographiques.** Elle permet d'exécuter des fonctions cryptographiques standard en assurant une protection ininterrompue des clés. En raison de sa protection conçue efficacement, zKeyBox est extrêmement difficile à manipuler et complique fortement toute tentative d'ingénierie inverse. zKeyBox utilise des **technologies brevetées** et a réussi un certain nombre d'audits de sécurité tiers.

Dans le cas d'applications logicielles existantes qui sont déjà associées à des modules cryptographiques, zKeyBox peut simplement remplacer ces derniers dans le code. Par conséquent, une application protégée par zKeyBox sera fonctionnellement équivalente à l'application d'origine et assurera une protection robuste de ses clés.

La procédure générale pour appliquer la protection zKeyBox est la suivante :

1. liez la bibliothèque statique zKeyBox à l'application cible que vous souhaitez protéger.
2. modifiez le code qui utilise les fonctions cryptographiques de bas niveau pour qu'elles utilisent l'API zKeyBox.
3. concevez et déployez votre application protégée par zKeyBox.

Figure 4: Applying zKeyBox protection to the target application

Fonctionnalités principales

ZKeyBox fournit une implémentation en boîte blanche pour un certain nombre d'algorithmes standard de l'industrie qui peuvent être exécutés sur un large éventail de plates-formes cibles comme le montrent les tableaux suivants :

Algorithmes pris en charge
Chiffrement
Déchiffrement
Signature
Vérification
Génération de clés
Enveloppement de clé
Désenveloppement de clé
Accord par clé
Résumés (hachage) de clés
Dérivation de clé

Plates-formes prises en charge
Android
iOS
tvOS
macOS
Windows
PlayStation
glibc/Linux
uClibc/Linux
musl/Linux
MinGW

Les chiffrements les plus populaires tels que AES, RSA, ECC, DES et Speck sont pris en charge. Étant donné que la liste exacte des algorithmes et des plates-formes cibles pris en charge est en constante évolution, veuillez consulter le guide de l'utilisateur de zKeyBox ou contacter votre responsable de compte Zimperium pour connaître les dernières fonctions et plates-formes prises en charge.

Security Aspects

Dans cette section, nous abordons certaines des caractéristiques de sécurité génériques de zKeyBox.

zKEYBOX™

Domaine crypté

Un domaine crypté fait partie d'un programme où toutes les données sont stockées sous forme chiffrée et toutes les opérations sont brouillées. En raison de la modification de la vitesse d'exécution impliquée (puisque le code brouillé entraîne nécessairement un perte de performances), un domaine crypté n'est généralement jamais utilisé pour un programme entier, mais plutôt pour ses parties essentielles telles que les algorithmes cryptographiques et le code qui gère les clés.

zKeyBox fournit un domaine crypté complet pour travailler avec des clés cryptographiques. La bibliothèque expose un ensemble de fonctions d'API à l'application appelante de manière à ce qu'il n'y ait pas de possibilité (et pas besoin) que l'application ou le pirate obtienne les clés en texte clair.

Le calcul dans un domaine chiffré est une caractéristique centrale de zKeyBox. Cela signifie que même lorsqu'un algorithme cryptographique est exécuté, les clés et autres données avec lesquelles il fonctionne ne sont jamais révélées en texte clair. En outre, toute tentative d'altération de l'algorithme ou de séparation des clés entraînera probablement un blocage de l'application.

Obfuscation

zKeyBox masque les clés secrètes et le flux d'exécution des algorithmes cryptographiques. Il est presque impossible de rétroconcevoir la logique et de suivre les étapes logiques. Étant donné que les outils de débogage standard ne fournissent pas de statistiques significatives aux pirates qui analysent le code de la boîte blanche unique, les méthodes d'altération traditionnelles sont inefficaces avec zKeyBox.

Par conséquent, même s'il existe des faiblesses théoriques découvertes dans les algorithmes cryptographiques de l'industrie mis en œuvre par zKeyBox, la nature brouillée du fonctionnement de la bibliothèque compliquera considérablement les attaques potentielles ou les rendra même impossibles.

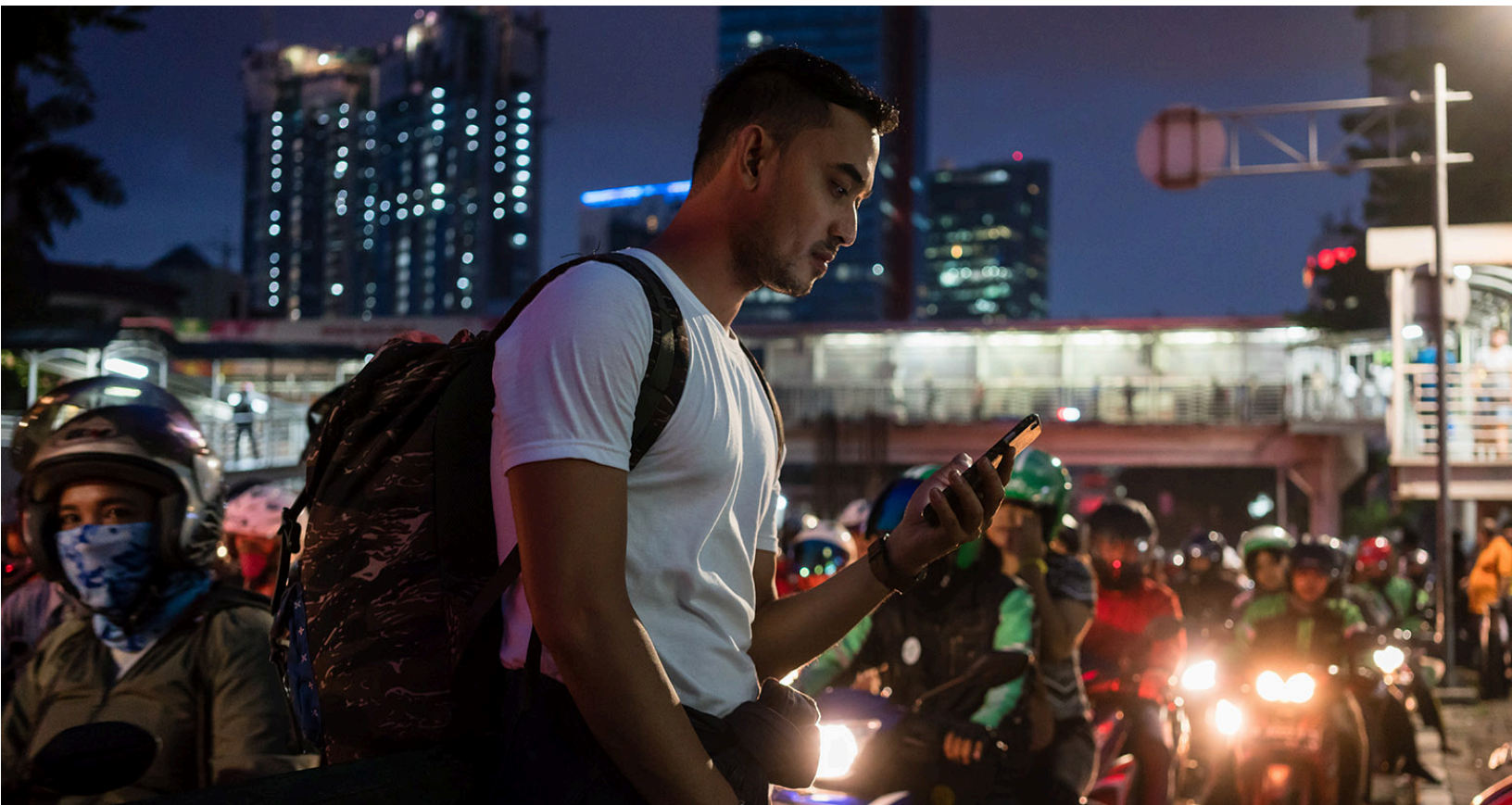
Diversification

La diversification logicielle est une méthode permettant d'ajouter la randomisation à un binaire exécutable et à ses données d'entrée et de sortie de sorte que les différentes instances du même logiciel semblent différentes dans tous les cas. La diversification logicielle contrecarre les tentatives d'exploitation des informations obtenues par un attaquant dans le cadre d'un déploiement pour en compromettre d'autres. Il est beaucoup plus difficile de développer un schéma de piratage universel pour des instances logicielles diversifiées, c'est-à-dire que chaque instance de logiciel doit être piratée individuellement.

La diversification fait partie intégrante de tous les produits Zimperium. ZKeyBox, en particulier, comporte un système de diversification à deux niveaux. Tout d'abord, le binaire de chaque instance zKeyBox est généré à partir d'une valeur initiale aléatoire qui assure la diversité du code, ce qui signifie que l'empreinte binaire de chaque application qui utilise la bibliothèque est unique, rendant la création d'outils de piratage universels presque impossible. Deuxièmement, chaque instance zKeyBox utilise un modèle différent pour crypter les clés qu'elle enregistre et charge à partir du stockage (diversité des données). Cela signifie que le pirate ne peut pas prendre la bibliothèque zKeyBox d'une application compromise et l'utiliser pour déchiffrer les clés d'autres applications.

Protection contre les attaques en boîte blanche

L'équipe de recherche de Secure Key Box teste et améliore constamment le produit afin de garantir la sécurité contre les attaques en boîte blanche connues. Un exemple est l'attaque Billet, qui est probablement la plus connue qui peut viser certains types d'implémentations AES en boîte blanche14. L'attaque dépend de certaines caractéristiques présentes dans la mise en œuvre spécifique AES en boîte blanche. Par exemple, il est supposé que l'implémentation en boîte blanche ressemble à une séquence d'applications S-box et de permutations des octets codés. Cependant, l'implémentation zKeyBox d'AES n'a pas les caractéristiques qui permettent l'application du type d'attaque spécifié. Par conséquent, l'attaque est rendue inutile contre zKeyBox.



Cas d'utilisation sélectionnés

Solution de paiement EMV tokenisée

Un cas d'utilisation typique de zKeyBox est de sécuriser des parties d'une solution de paiement EMV tokenisée sur un dispositif mobile. Les principales fonctions de ces systèmes incluent le provisionnement des dispositifs, le provisionnement des tokens, le stockage des données des tokens, et le traitement des tokens.

Le provisionnement des dispositifs implique la création d'une identité de l'appareil mobile et sa mise en relation avec l'identité du titulaire de la carte dans l'écosystème de paiement. Au cours de ce processus, le dispositif acquiert une clé unique liée à l'identité du titulaire de la carte (telle que connue par l'émetteur de la carte). Le provisionnement des dispositifs peut utiliser un système d'accord de clé entre l'appareil et le serveur, une dérivation de clé, une signature numérique pour l'authentification, un chiffrement/déchiffrement pour le trafic de session, ainsi qu'un stockage sécurisé et lié au dispositif de la clé acquise. Le provisionnement des tokens demande et reçoit les tokens à usage unique pour une utilisation ultérieure dans les transactions de paiement. Au cours de ce processus, une signature numérique est utilisée pour l'authentification, et le chiffrement et le déchiffrement sont utilisés pour le trafic de session et la protection des données des tokens pendant le transit et le repos sur le dispositif. Le traitement du token se produit pendant le paiement lorsqu'il est utilisé (en remplacement du numéro de carte PAN) ; il implique le décryptage des données du token, le calcul de la valeur d'authentification (Retail MAC) et le cryptage des données modifiées du token.

Comme on peut le constater, un grand nombre d'opérations cryptographiques sont impliquées dans ce cas d'utilisation. Toutes ces opérations sont prises en charge par zKeyBox tout en garantissant que les clés et autres secrets impliqués ne sont jamais révélés sous une forme non chiffrée. Cela permet le déploiement de l'application de paiement sur des dispositifs qui ne prennent pas en charge l'environnement de sécurité matérielle et sur les ceux où cet environnement n'est pas disponible pour les développeurs.

Système de gestion des droits numériques

Les sociétés de divertissement et de médias du monde entier ont augmenté leur valeur grâce à des services de diffusion en continu innovants, des programmes, des concerts en direct, des entretiens quotidiens en coulisses, des émissions sportives et une variété d'événements musicaux et d'actualités qui peuvent être visionnés sur des appareils mobiles. Plus important encore, les consommateurs peuvent désormais consulter des contenus de divertissement spécifiques sur leurs propres dispositifs, où qu'ils se trouvent, notamment les avions, les taxis et autres moyens de transport en commun. Pour protéger le contenu contre le vol, des systèmes de gestion des droits numériques (DRM) doivent être en place et des solutions d'application de sécurité mobile sont nécessaires pour protéger les applications des joueurs elles-mêmes.

Étant donné que les systèmes DRM impliquent plusieurs opérations cryptographiques et dépendent de l'intégrité des clés cryptographiques, les développeurs doivent ajouter un niveau de protection à leurs applications DRM pour empêcher les pirates informatiques de pénétrer le système DRM ou de voler les clés secrètes. zKeyBox est un outil idéal à cette fin, car il prend en charge tous les algorithmes cryptographiques standard de l'industrie utilisés dans les solutions DRM et ne révèle jamais les clés cryptographiques sous une forme non chiffrée.

Étapes suivantes

Ce livre blanc a présenté un examen approfondi de zKeyBox par Zimperium. Nos mécanismes de protection de pointe vous aideront à protéger vos clés cryptographiques contre les attaques ainsi que les ressources les plus importantes pour vous et vos clients.

Contactez-nous pour savoir comment Zimperium peut vous aider à protéger vos clés cryptographiques, obtenir une démonstration, lancer un essai gratuit ou accéder à de plus amples informations.

À propos de Zimperium

Zimperium secures mobile devices and mobile applications so they can safely access sensitive data and systems. We are an advanced machine learning-based solution with a privacy focus, supporting iOS, Android, and ChromeOS platforms.

Zimperium sécurise les appareils mobiles et les applications mobiles afin qu'ils puissent accéder en toute sécurité aux données et systèmes sensibles. Nous offrons une solution avancée basée sur l'apprentissage machine, axée sur la confidentialité, prenant en charge les plates-formes iOS, Android et ChromeOS.

La Mobile Application Protection Suite (MAPS) de Zimperium aide les entreprises à créer des applications mobiles sécurisées et conformes. Il s'agit de la seule solution unifiée qui combine une protection complète intégrée à l'application et une visibilité centralisée des menaces.

- Notre protection intégrée à l'application inclut le blindage des applications, l'auto-protection des applications d'exécution côté client (RAPA) et des techniques de sécurisation contre les programmes malveillants.

- Notre visibilité permet de tester en continu la sécurité des applications (AST) au cours du développement et de la visibilité de l'exécution concernant les menaces et les attaques.

Sources

1. https://www.documentcloud.org/documents/3010178-Volkswagen-amp-HiTag2-Keyless-Entry-System.html
2. https://marcan.st/uploads/25c3_console_hacking
3. G. Lowe, "An attack on the Needham-Schroeder public key authentication protocol", Information Processing Letters, Volume 56, Issue 3, 1995
4. S. Fluhrer, I. Mantin, A. Shamir, "Weaknesses in the Key Scheduling Algorithm of RC4", Selected Areas in Cryptography. SAC 2001. Lecture Notes in Computer Science, vol 2259, 2001
5. http://heartbleed.com
6. A. Shamir, N. van Someren, "Playing Hide and Seek With Stored Keys", Financial Cryptography. FC 1999. Lecture Notes in Computer Science, vol 1648, 1998
7. https://github.com/mmozeiko/aes-finder
8. https://tools.ietf.org/html/rfc2631
9. P. Kocher, J. Jaffe, B. Jun, "Differential Power Analysis", CRYPTO '99 Proceedings of the 19th Annual International Cryptology, Conference on Advances in Cryptology, 1999
10. M. Hutter, J. Schmidt, "The Temperature Side Channel and Heating Fault Attacks", CARDIS, 2013
11. J. A. Halderman et al, "Lest We Remember: Cold Boot Attacks on Encryption Keys", Proc. 17th USENIX Security Symposium (Sec '08), 2008
12. https://meltdownattack.com
13. B. Wyseur, "White Box Cryptography", PhD thesis, 2009
14. O. Billet, H. Gilbert, C. Ech-Chatbi, "Cryptanalysis of a White Box AES Implementation", Selected Areas in Cryptography. SAC 2004. Lecture Notes in Computer Science, vol 3357, 2005

ZIMPERIUM®

zimperium.com
844.601.6760 | info@zimperium.com

Zimperium, Inc
4055 Valley View, Dallas, TX 75244