Rapport Malware

CHAPELLE Théo - LATRACHE Hind - THIEBAUX Valentin

18 Décembre 2023

1 Introduction

Le malware que nous avons codé est un malware de type B, c'est-à-dire un malware ne contenant pas de clé prédéfinie à l'exécution.

2 Obfuscation et astuces

2.1 Fonctions

Pour cacher certains appels de fonctions, et surtout pour faire croire qu'une clé existe, nous avons réalisé un script python créant 10.000 fonctions de façon aléatoire avec des paramètres, des types de retour et des valeurs aléatoires. Ces fonctions sont appelées plusieurs fois dans le "vrai" code pour effectuer des calculs inutiles et/ou faire croire à des tests sur la clé directement.
Dans les fonctions principales, nous avons réalisé une fonction appliquant un XOR entre deux paramètres et un chiffrement par décalage (César) appelé avec un nombre (une clé) permettant de faire la permutation. Ces deux fonctions sont utilisées sur la clé et certains calculs pour faire croire à une modification de la clé donnée en argument.
Ces fonctions ont été empoisonnées par les fonctions aléatoires pour faire croire qu'elles sont importantes.

2.2 Test sur les fonctions

Certaines fonctions, en particulier celles utilisées pour effectuer des calculs spécifiques sur la clé, ont été testées pour vérifier qu'elles n'ont pas été modifiées. Pour cela, nous avons utilisé une boucle qui parcourt la fonction et qui calcule la somme des instructions présentes (si l'instruction change, le nombre correspondant change également). Nous avons également fait le test de la fonction `IsDebuggerPresent()` à plusieurs endroits dans le code et dans les fonctions pour tester si les défenseurs utilisaient un débogueur.

La majorité des tests et des actions sur les fonctions sont en réalité présentes pour dissimuler des calculs sur notre clé factice. Cela sert donc à cacher une clé potentiellement prédéfinie, bien que notre malware n'en possède pas.

2.3 Auto-modification

Pour rendre plus difficile l'analyse du code et perturber la compréhension du fonctionnement de notre programme, nous avons utilisé la technique d'auto-modification dans plusieurs fonctions. Parmi ces fonctions sont celles de chiffrement, notamment la fonction de chiffrement par décalage (César), la fonction qui traite la musique de Mario et la fonction de chiffrement XOR.

3 Anti-debug

Dans le malware, si l'utilisateur ne rentre pas un argument attendu ou si il utilise un débogueur, alors nous lançons une musique de Mario.

Cette musique a été réalisé par combinaison des fonctions `Sleep` et `Beep`, toutes les deux cachées par des automodifications. La musique de Mario Bros. en 8bit est donc jouée à chaque détection de debug, ce qui arrive extrêmement souvent grâce aux fonctions générées. Nous espérons avoir pu ralentir l'adversaire à la compréhension grâce à cela.

Egalement, si nous avons préféré appliquer un traitement plus "doux" que de détruire la machine virtuelle, c'est pour nous démarquer. En réalité, il est tout aussi pénible d'écouter la musique de Mario en boucle (d'autant qu'avec les appels à Sleep, ce n'est pas esquivable) que de devoir remettre au point une VM à chaque fois.