Emy LIEUTAUD - Yunqiao ZHANG - Gautier KLAM

Fonctionnement de notre Malware

Pour faire fonctionner notre malware il suffit de double cliquer sur l'exe puis entrer la clé.

Clé : a30f65aa6aac13f24c3f8252a89fbd2ee46cfea2afbfc6cc06d889

Dans un premier temps, nous avons obfusqué la clé :

En premier, nous avons converti des lettres de la clé par des nombres :

- a = 78
- b = 24
- c = 96
- d = 3
- e = 51
- f = 333

La nouvelle clé est désormais :

N30M65NN6NN`13M24`3M8252N89M↑♥23346`M3N2NM↑M`6``06♥889

Cette nouvelle clé, nous l'avons converti en binaire :

010011100011001100110000010011010011011000110101010011100100111000110110010
011100100111001100000001100010011001101001101001100100011010001100000001100
110100110100111000001100100011010100110010010011100011100000111001010011010
001100000000011001100100011001100110011001101000011011001100000010011010011
001101001110001100100100111001001101000110000100110101100000001101100110000
001100000001100000011011000000011001110000011100000111001

Cette clé on l'a encodé en XOR encryption avec la clé 0x5A.
La nouvelle clé est :

jkjjkkkjjjkkjjkkjjkkjjjjjkjjkkjkjjkkjkkjjjkkjkjkjkjjkkkjjkjjkkkjjjkkjkkjjkjjkkkjjkjjkkkjjkkjjjjjjkkjjjkjjkkjjkkjkjjk
kjkjjkkjjkjjjkkjkjjjkkjjjjjjjkkjjkkjkjjkkjkjjkkkjjjjjkkjjkjjjkkjkjjkkjjkjjkkkjjjkkkjjjjjkkkjjkjkjjkkjkjjjkkjjjjjjjjk
kjjkkjjkjjjkkjjkkjjkkjjkkjjkkjkjjjkkjkkjjkkjjjjjkjjkkjkjjkkjjkkjkjjkkkjjjkkjjkjjkjjkkkjjkjjkkjkjjjkkjjjkjjkkjkkjjj
jjjjkkjkkjjkkjjjjjkkjjjjjjkkjjjjjkkjkkjjjjjjkkjjkkkjjjjjkkkjjjjjkkkjjk

Méthode pour comparer la clé saisie par l'utilisateur avec notre clé : on encode la clé entrée par l'utilisateur.

Emy LIEUTAUD - Yunqiao ZHANG - Gautier KLAM

Dans un second temps, nous avons chiffré les différents messages :

Pour le premier message :

“La cle est invalide : elle doit etre inferieure ou egale à 64 caracteres.”
Pour le crypter, le code ASCII de chaque caractère est ajouté de manière répétée au nombre dans "4962873" et si la plage dépasse 032 (espace) à 122 ("z") du code ASCII, l'opération modulo est effectuée.

Le décryptage et le cryptage se font dans l'ordre inverse, il suffit de faire le message chiffré moins le nombre dans la boucle.

Pour le second message :

“Bravo ! Tu as trouvé “
Nous l’avons crypté en XOR cypher avec la clé : Cpbrj&&!Vv$du'uplqsc'
Ainsi pour la suite du message on a : LACLESECRETE
Que l’on a crypté à l’aide du chiffrement de vigenère qui a pour clé : MALWARE. C’est un chiffrement par substitution polyalphabétique dans lequel une même lettre du message clair peut, suivant sa position dans celui-ci, être remplacée par des lettres différentes.

Pour le dernier message :

“La clé est invalide : elle doit contenir des caractères hexadécimaux.”
On l’a crypté par le chiffrement de César avec la clé égale 6. Le chiffrement de César consiste à décaler d’un certain nombre les lettres de l’alphabet. Ici le a devient g.

Pour finir, nous avons obfusqué notre débugger :

La fonction IsDebuggerPresent() est remplacée par la fonction printf(“”) à l’aide de VirtualProtect.

Le message “Un debugger est present” est crypté en ROT13. ROT13 est un chiffrement de césar avec un décalage de 13 caractères.