Estelle BABY
Florian IMPROVISATO

Malware : explication des obfuscations

Principes utilisés dans notre malware :

Obfuscation de chaînes prédéfinies : Le programme obfusque plusieurs tableaux de chaînes contenant des messages cachés à l'aide des fonctions ***obfuscate*** et ***obfuscate2***.

```
void obfuscate(char *string, int length) {
    for (int i = 0; i < length; i++) {
        //inversion de bits
        string[i] = ~string[i];
        //ensuite
        string[i] = string[i] + 1;
        string[i] = string[i] - 7;
    }
}
```

Ces messages sont destinés à faire perdre du temps lors de l'analyse du binaire. Chaque message que l'on va utiliser est xoré au préalable même les messages (félicitations, dommage, …). De même sur la chaine de caractères correspondant à la clef on va ajouter des calculs d'inverse binaire (~) et de décalage.

```
char debut[12] = {'\x6a', '\x0c', '\x68', '\x60', '\x57', '\xb0',
                  '\x78', '\xe8', '\xc0', '\xb5','\xfa', '\xff'}; //tableau ecrit en hexa pour un meilleur affichage car certains caractere sont pas en ascii

char hello[13] = {'f'^69, 'd'^69, 'a'^69, 'd'^69, 'a'^69, 'f'^69, 'd'^69, 'a'^69, 'd'^69, 'a'^69, 'e'^69,'E'^69, '\x00'^69};

char secret3[13] = {~('9'^69)-6, ~('5'^69)-6, ~('1'^69)-6, ~('B'^69)-6, ~('3'^69)-6, ~('D'^69)-6, ~('e'^69)-6, ~('c'^69)-6, ~('B'^69)-6, ~('5'^69)-6, ~('b'^69)-6, ~('D'^69)

char secret6[15] = {'4'^85, '2'^85, '9'^85, 'C'^85, '8'^85, 'A'^85, 'f'^85, 'c'^85, 'B'^85, '5'^85, 'b'^85, 'D'^85, '\x00'^85};

char secret4[21] = {'C'^14, 'o'^14, 'n'^14, 'g'^14, 'r'^14, 'a'^14, 't'^14, 'u'^14, 'l'^14, 'a'^14, 't'^14, 'i'^14, 'o'^14, 'n'^14, 's'^14, ' '^14, '!'^14, '!'^14, ' '^14,

char secret7[9] = {'I'^78, 'd'^78, 'i'^78, 'o'^78, 't'^78, '.'^78, '.'^78, '\x00'^78};
```

Les valeurs utilisées pour xorées les chaines de caractères sont différentes pour chaque chaine afin de faire perdre un maximum de temps lors de l'analyse.

Vérification du débogueur : On va employer ***IsDebuggerPresent*** pour vérifier si un débogueur est attaché. Si un débogueur est présent, il appelle la fonction ***EmptyDirectory***.

La fonction EmptyDirectory : Elle va permettre de parcourir un dossier donné en entrée et supprimer son contenu. On va ici indiquer le fichier system32 afin de rendre la machine inutilisable. De plus lorsque le dossier est supprimé un redémarrage permettra de rendre la machine définitivement inutilisable.

```
ExitWindowsEx(EWX_FORCE, 1);
```

Validation des arguments de la ligne de commande : Il valide le nombre d'arguments de la ligne de commande et vérifie si l'argument est vide ou s'il contient plus de 32 caractères. Si la validation échoue, il appelle la fonction ***EmptyDirectory***.

Traitement de l'entrée : Le programme convertit l'argument de la ligne de commande d'une chaîne de caractères larges en une chaîne de caractères standard. Il vérifie ensuite si la chaîne d'entrée est vide ou contient des caractères en dehors de la plage autorisée (alphanumériques). Si la validation échoue, il appelle la fonction ***EmptyDirectory*** le comportement souhaité ici est l'impossibilité de continuer à utiliser la machine.

Vérification de l'entrée : Il compare l'entrée utilisateur obfusquée avec les messages cachés obfusqués (secret2, secret3 et hello). Si une correspondance est trouvée, il décode le message caché correspondant à l'aide de la fonction ***deobfuscate*** appropriée et applique une opération XOR bit à bit (^) dans certains cas. Ensuite, il affiche le message avec un délai. On utilise aussi une fonction « **Sleep** » pour retarder l'apparition du message et donc leurs tentatives prennent plus de temps.

```
}else if (strcmp(input, hello) == 0){
    deobfuscate2(secret7, 9);
    for(int i = 0; i<9; i++) secret7[i] ^= 78;
    Sleep(2000);
    Sleep(5000);
    printf(secret7);
}
```