Sécurité des Applications

Octobre 2020 | **Serie 4**

Stéphane Küng

1 Administratif

Ce travail **individuel** doit être rendu avant la date indiquée sur le ScoreBoard au format **PDF** sur CyberLearn.

2 Exercices 4

2.1 MEM1 - AoE 1

Le jeu Age of Empire trop est difficile voir impossible. Utiliser un éditeur de mémoire pour obtenir un chateau. Documenter comment vous avez fait, les outils utilisés, et ce que vous avez modifié. Mettez votre code chateau dans le rapport `PDF`

2.2 MEM2 - AoE 2 (Dificile)

Le jeu Age of Empire trop est difficile voir impossible. Coder, avec les fonctions vues en cours (ptrace) un programme (souvent appellé `trainer`) vous permettant d'automatiser la modification des mémoires du jeu. Documenter votre analyse et envoyer votre code source joint au rapport `PDF`

2.3 EDIT1 - AoE 3

Modifier le jeu Age of Empire de sorte qu'il distribue plus de ressource lorsque l'utilisateur en demande. Le jeu doit continuer à fonctionner.

2.4 EDIT2 - CrackMe 4

Décompiler le CrackMe 4 pour le comprendre et modifiez le si besoin. Donner votre solution, celle-ci doit fonctionner sur la version **non modifiée** du binaire

NOTES : Vous pouvez utiliser le logiciel `ILSpy` pour la modification du binaire.

2.5 SIG1 - Signature 1

Il vous faut d'abord modifier le CrackMe 4 de sorte que votre prénom soit la solution du binaire. il vous faudra ensuite créer un certificat ou une PKI et signer le `CrackMe4.exe`. Incluez le timestamp dans la signature.

Reportez dans votre rapport la modification de l'Exe, comment les certificats ont été générés et comment vous avez signé le binaire avec le timestamp et joignez votre `.exe` signé au rapport `Zip/7z`

NOTES : Vous pouvez utiliser le logiciel `SignTool` de Microsoft pour la signature, et `ILSpy` pour la modification du binaire.

2.6 SIG2 - Signature 2

Ecriver un Hello World en `PowerShell` et signez votre script de la même manière que l'exercice précédent.

Expliquez comment vous avez signé le script avec le timestamp, ce qui change dans le script, et joignez le au rapport `Zip/7z`

2.7 SIG3 - Signature 3 (Exploratoire)

Sous Linux cette fois, trouvez une manière de signer un binaire de sorte que la signature soit **incluse** dans celui-ci. Il faut également avoir une façon de vérifier la signature contenue dans le binaire. Tentez de modifier le binaire et et confirmer que la signature ne fonctionne plus.

2.8 OBF1 - Obfuscation de binaire

Obfusquez le code du sous-marin donné dans la série 3. Selon les choix suivants :

- Modification du flow d'execution du binaire (avec LLVM Obfuscator par exemple). Documentez l'arbre des blocs d'intrustions avant et après les passes d'obfuscation (avec des capture).
- Commpressez votre code avec un outil comme UPX par exemple. Documentez

NOTE : Autres outils que proposé **+1pts**

2.9 TIME1 - TimeAttack On CrackMe 5

Le CrackMe 5 est vulnerable à une timming attack. Le but de cet exercice est de coder un script ou application qui teste toutes les combinaisons possibles. Le CrackMe vérifie chaque caractère dans l'ordre, si un cactatère est bon, il passe au suivant. Testez toutes les possibilités sur le premier caractère puis une fois trouvé passez au suivant.

NOTE : la commande `time` peut être utile.