Cégep du Vieux Montréal
25 mai 2021

Ver informatique

Sprint 3 – Rapport de Projet

Réalisé par
Karl Boulais

Présenté à
Pierre-Paul Monty

Dans le cadre du cours
Projet synthèse
420-B65-VM gr. 00001

Table des matières

Le projet

Retour général

Le but de ce projet était de créer un ver informatique qui se transmettrait et se reproduirait par ses propres moyens d'un système Windows Server 2016 à un autre en abusant de la faille de sécurité Eternal Blue (MS17-010).

Nous avons ajouté une composante de centre de contrôle qui permet d'avoir une base de données des serveurs infectés ainsi qu'une interface par laquelle nous pourrions contrôler une de ces machines.

Résumé du développement

Difficultés

Décider d'abuser d'une faille de sécurité n'était pas chose facile. Nous savions que nous aurions des embuches sur l'application et la mise en place et il s'en est avéré.

Tout d'abord, la faille de sécurité sur les ordinateurs Windows est désormais corrigé. En pratique, ça veut dire que pour développer un programme autour de cette faille, il faut user de machines virtuelles puisque les anti-virus éliminent tous les programmes et code connexe qui pourrait servir à l'exploiter. De plus, même en installant un système d'exploitation passé date sans les corrections appliquées, celui-ci peut se mettre à jour lui-même. En somme, nous devions nous connecter sur une machine virtuelle à distance et couper l'accès à internet.

De plus, nous pensions pouvoir utiliser un dépôt git et le greffer à notre projet, malheureusement le code n'est plus fonctionnel et certainement pas mis-à-jour. Donc nous avons abandonné l'exploitation de la faille et conséquemment la partie vers informatique de notre projet.

Réussites

Heureusement, nous avons réussi, nous croyons à avoir développé un modèle de commande et contrôle à distance intéressant, quoique quelque peu rudimentaire. Essentiellement, le deuxième aspect du projet était qu'une fois que nous nous étions implantés dans un système nous puissions le contrôler et nous avons réussi sur ce plan.

Donc nous avons trois parties au projet : Le serveur qui reçoit les connexions clients et les clients qui s'y connectent et une base de données. Le serveur écoute sur deux ports soit le 6666 pour les connexions Reverse Shell et le 9999 pour les connexions de transmission de données.

Fonctionnalités

Parfaitement fonctionnel

- La base de données
- Le serveur multithread de connexion TCP qui permet de recevoir des connexions Reverse Shell et de réception de données.
- Interface de contrôle en ligne de commande

Semi fonctionnel

- La reproduction fonctionne, mais est limité, puisque la transmission est impossible.
- Installation et persistance dans un système en s'installant en tant que service Windows.

Non fonctionnel

- Intégration du module Metasploit/Dépôt git

Abandonné

- Interface graphique pour contrôler l'aspect serveur en Qt
- Utilisation de Ansible pour gérer la production de masse de machines virtuelles

Amélioration possible

Avec plus de temps nous pourrions pousser beaucoup plus loin, les possibilités de ce projet sont presque infinies! Faire un outils de contrôle à distance et «monitoring» nous vient en tête.

Ajouter une interface graphique. Ajouter au module programme un module qui permet d'extraire les données du presse-papier lorsqu'il change.

Auto-évaluation

Je crois que la planification de ce projet était bonne. Je suis un peu déçu de la faille qui ne fonctionne pas, sinon, je pense que le potentiel est là, mais encore trop brut pour être apparent à l'œil externe. Donc que penser de tout ça? Je crois que j'ai fait un bon travail avec les connaissances que j'avais. Nous avons repoussé nos limites implémentant le «multithreading» et en construisant de toutes pièces un programme qui peut contrôler des ordinateurs à distance.