Cégep du Vieux Montréal
26 février 2021

Ver informatique

Sprint 1 - Conception

Réalisé par

Karl Boulais

Présenté à

Pierre-Paul Monty

Dans le cadre du cours

Projet synthèse
420-B65-VM gr. 00001

Table des matières

Présentation générale du projet

Créer un ver informatique qui se transmet et se reproduit par la réimplantation d'une faille de sécurité dans le protocole SMBv1. Nous démontrerons que la réplication du ver donne une capacité de transmission supérieure (voire exponentielle) à celle d'une infection sérialisée à partir d'un point unique.

Présentation précise du projet

Constitution du projet

Le projet aura deux programmes principaux :

- Centre de commande et de contrôle :
  - Liste de machine infectées
  - Information pour donner une capacité de contrôle à distance
- Ver :
  - Version infectant d'un point central
  - Version infectant à partir de toutes les nouvelles infections.

Plateforme et terrain de jeu

Le projet utilisera le système d'exploitation Windows server Core comme terrain de jeu et sera virtualisé sur la plateforme Proxmox. L'environnement sera mis sur pied en utilisant Ansible un programme de déploiement multi nœud, ce qui nous permettra de faire la mise en place dans un premier cas de 100 machines virtuelles et dans un deuxième cas de 1000 machines virtuelles.

Objectifs

Nous voulons utiliser une vulnérabilité de système affectant le protocoles SMBv1 nommée Eternal Blue et utilisé celle-ci pour se distribuer dans un sous domaine peuplé de serveur Windows. Pour ce faire, nous écrirons un ver informatique en C++ qui se reproduira et infectera d'autre machine à sa portée. Une fois installé le programme ouvrira le port 9999 et écoutera des

commandes du centre de contrôle, il enverra aussi son statut et les informations du système infecté.

Du point de vue du centre de contrôle, nous aurons une interface graphique minimaliste codé à l'aide du module Qt qui facilitera la représentation et les interactions avec les machines infectées. Nous aurons le nom des machines, leur adresse IP, les noms d'utilisateurs et un lien pour ouvrir une connexion à distance RDP.

Embûches potentielles

Cette faille de sécurité est très complexe. Nous ne prétendons comprendre toutes les finesses de cette dernière, mais heureusement le groupe RiskSense a écrit un module pour la plateforme Metasploit et met à disposition des outils pour reproduire l'abus de la faille. Nous utiliserons donc leur plateforme pour abuser de la faille, mais nous développerons tout le reste. C'est-à-dire l'élaboration du centre de contrôle et du ver.

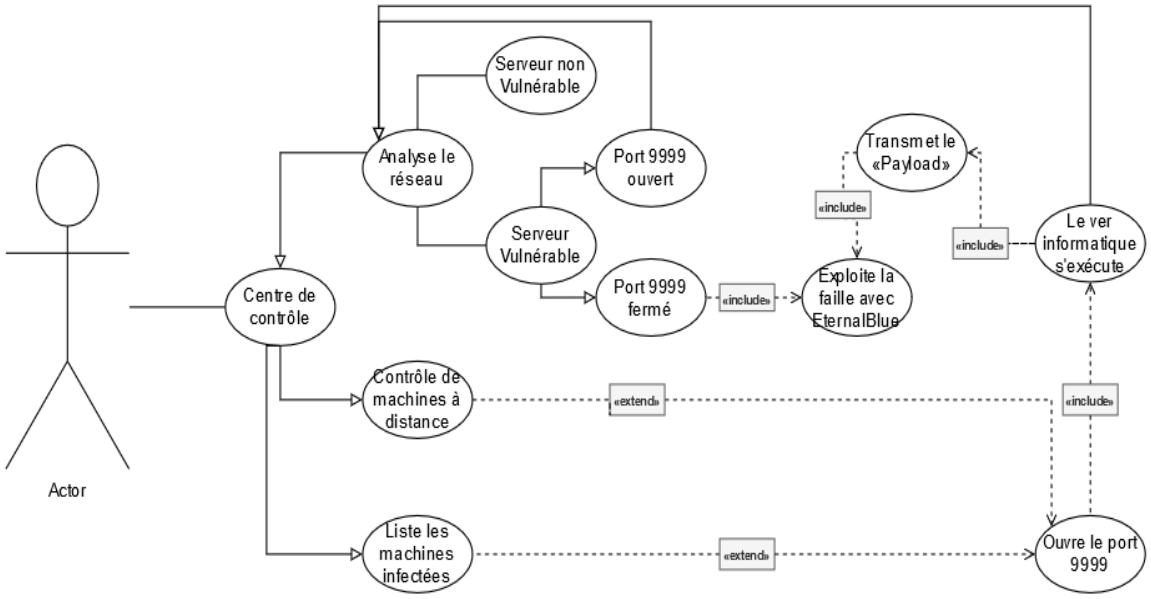
Une autre embûche potentielle est l'élaboration de connexion réseau en c++ et le *hook* avec la composante Metasploit. Effectivement, il serait beaucoup plus aisé de le faire avec un langage de plus haut niveau tel que python disposant de plusieurs mécanisme et librairie facilitant ce genre de manipulation. Mais nous croyons que de pouvoir déployer un exécutable précompilé facilitera la partie de transmission.

Présentation des patrons de conception envisagés

- Adaptateur :
    - Interagir avec une plaforme codée en Ruby au travers de c++
- MVC :
    - Représentation du centre de contrôle

Les aspects techniques de la conception

Cas d'usage



Conception des interfaces usagers

Centre de commande

File	Edit	Options	Tools		Help
Serveur infectés					
ip	HostName	Status	Remote	More	Destroy
172.0.0.10	svr10	Up	RDP	Info	Destroy
172.0.0.11	svr11	Up	RDP	Info	Destroy
172.0.0.12	svr12	Up	RDP	Info	Destroy
172.0.0.13	svr13	Up	RDP	Info	Destroy
172.0.0.14	svr14	Down	RDP	Info	Destroy
172.0.0.15	svr15	Up	RDP	Info	Destroy
172.0.0.16	svr16	Up	RDP	Info	Destroy
172.0.0.17	svr17	Down	RDP	Info	Destroy

Diagramme de classe UML

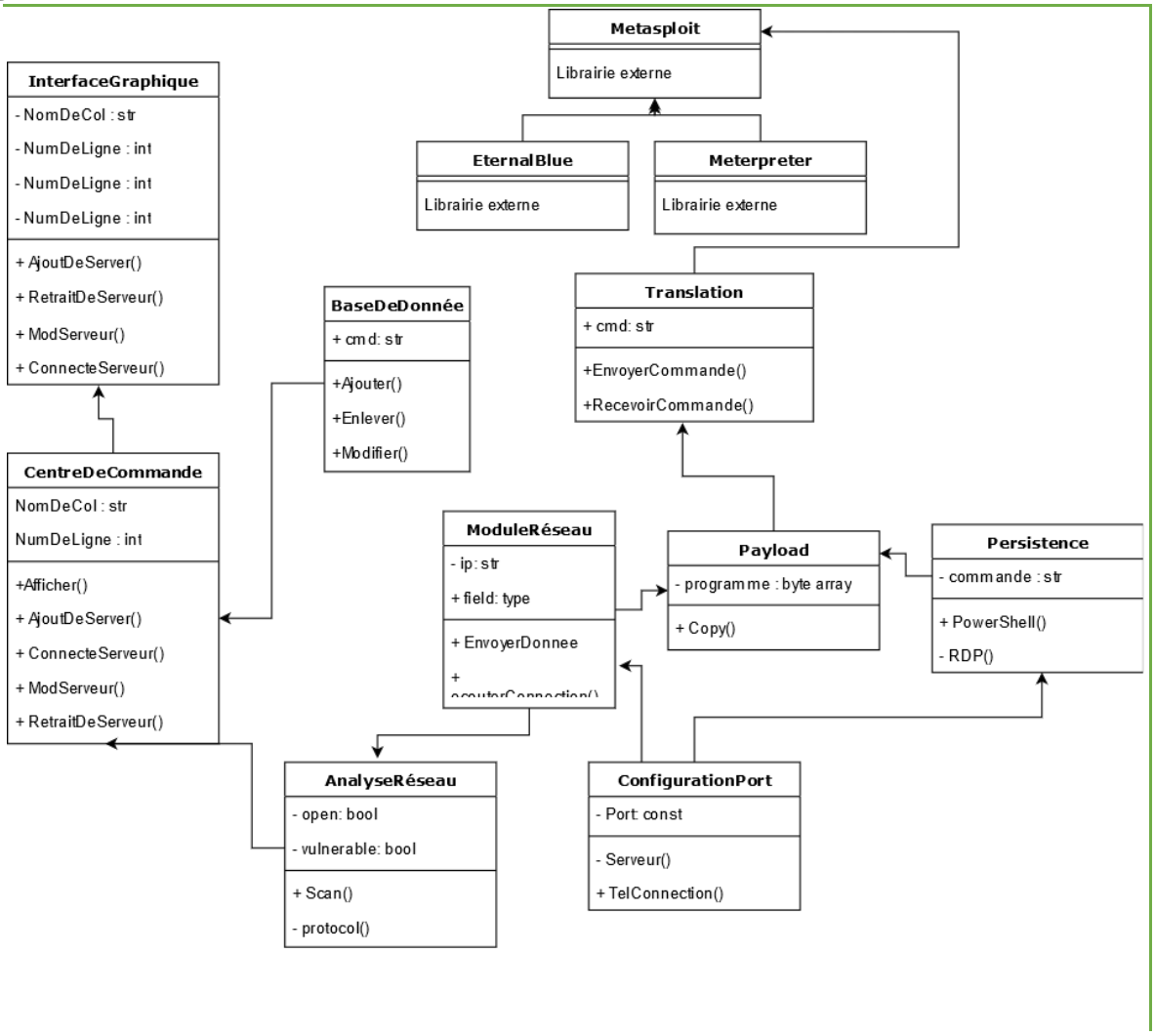


Diagramme de réseau