Rapport de synthèse

15/06/2017

Étudiants
BOUTANT Thomas
SALMAN Alexandre

Encadrants
CHARMET Fabien
BLANC Grégory

Sommaire

I - Contexte du projet

A - Cadre général

En 2ème année à Télécom SudParis, les étudiants ont l'opportunité de travailler sur un projet industriel, de recherche ou de développement : c'est le projet Cassiopée. Parmis les projets proposés en cette année 2016/2017, un tournait autour du thème de la sécurité informatique : notre projet, le n°37.

La sécurité informatique, un domaine très demandé

En 2015, selon le CESIN (le Club des experts de la sécurité de l'information et du numérique), 81% des entreprises françaises ont été visées par une cyberattaque, profitant du manque de préparation des entreprises à prévenir ou répondre aux incidents. Donc, parce que le monde d'aujourd'hui et de demain utilise très largement les nouvelles technologies, il est nécessaire et même vital pour les entreprises de sécuriser leurs outils de travail.

De nos jours, il y a plus d'offres que de demandes, ce qui peut s'expliquer par la nouveauté du besoin et par le manque de formation d'éventuels candidats. Pour débuter dans ce domaine, ce projet Cassiopée nous a paru idéal, puisqu'il nous permettait de créer les attaques et donc de pouvoir les reconnaître si celle-ci nous survient à l'avenir.

Le rendez-vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information (RESSI)

Cette année a eu lieu la 3ème édition de RESSI.

RESSI a pour objectif de rassembler une communauté francophone la plus large possible autour du thème de la sécurité des systèmes d'information et de catalyser les synergies entre les différents acteurs concernés par ce domaine : chercheurs, industriels soutenant une activité dans ce domaine, utilisateurs finaux de technologies de sécurité, gouvernementaux, enseignants et étudiants.

Une tradition s'est installée au fil des années avec la présence d'un challenge de sécurité informatique, permettant aux participants de tester leurs connaissances dans ce domaine.

B - Changement de projet

Nous avons commencé le projet “Challenge RESSI 2017” très exactement le 13/03/2017, alors que le projet Cassiopée a démarré début février. Cela s’explique par une nouvelle proposition de notre tuteur qui s’est accompagné d’un nouveau projet.

01/02 - 13/03 : Amélioration de la plateforme de génération d’épreuves “*Capture the Flag*” (CTF)
13/03 - 18/05 : Challenge RESSI 2017

Projet initial : Amélioration de la plateforme de génération d’épreuves CTF

Dans celui-ci, nous devions créer des épreuves simples de cryptographie, de stéganographie, de réseau, de hacking, de logique, de web client, de web serveur, etc. Pour ce faire, nous utilisions comme exemple des sites de référence dans le domaine, tels que RootMe, NewbieContest, OverTheWire.

Dans ces épreuves, l’objectif est de trouver un mot-clé. Avec la plateforme de génération d’épreuves, notre but était de pouvoir changer le mot-clé à trouver. Illustrons notre propos : ce serait comme si nous laissions un message à un endroit et qu’on construisait un labyrinthe autour. Si on veut changer le message, on écraserait alors le labyrinthe par un nouveau mais avec le nouveau message.

En 1 mois, nous avons créé chacun une épreuve de cryptographie (substitution).

C - Objectifs du nouveau projet : RESSI

Notre contribution au projet aura été de participer à la création du challenge. Nous avons créé l'application de e-commerce, le serveur web, le scoreboard, certaines vulnérabilités et les scripts d'achat. Nous avons également implémenté dans notre machine d'autres vulnérabilités qui nous avaient été fournies par des membres du comité d'organisation du challenge.

Nous sommes également allés à Autrans, près de Grenoble, pour aider à l'organisation du challenge. Nous étions responsables de la partie achat et scoring, et nous contribuions aux attaques.

II - Organisation du projet

A - L'équipe

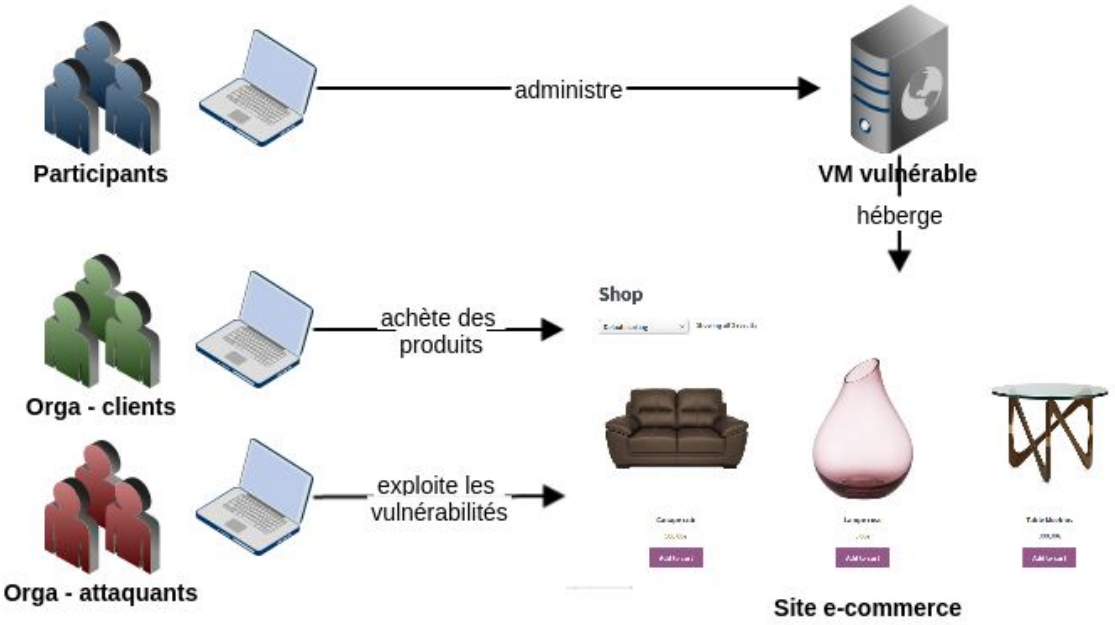
L'équipe qui travaillait sur le projet "Challenge RESSI 2017" était constituée de nous deux, de M. Blanc notre encadrant, ainsi que de deux membres du comité d'organisation du challenge, M. Nicomette et M. Lalande. Dans un premier temps, nous nous sommes tous les deux intéressés à la création du site, puis Thomas s'est occupé davantage de l'aspect applicatif (scripts d'achat, vulnérabilités applicatives), tandis qu'Alexandre s'est occupé des autres vulnérabilités (vulnérabilités système, implémenter les vulnérabilités créées par M. Nicomette) ainsi que de la création du scoreboard.

B - Organisation du projet

En terme de livrables, nous fournissions chaque semaine, le mercredi ou le jeudi, un compte rendu de notre avancement. La majorité du travail se faisait le lundi après-midi et le mardi après-midi, bien que dans la dernière phase du projet notre emploi du temps ait été un peu plus chaotique. Nous avions en moyenne une réunion par semaine avec notre encadrant, mais ce n'était pas figé : nous venions le voir lorsque nous avions besoin d'aide ou qu'il avait une nouvelle tâche à nous donner. Nous avons également participé à plusieurs appels groupés avec les autres membres du projet, pour rendre compte de l'avancement et discuter de nouvelles vulnérabilités à ajouter.

III - Le challenge

A - Le principe

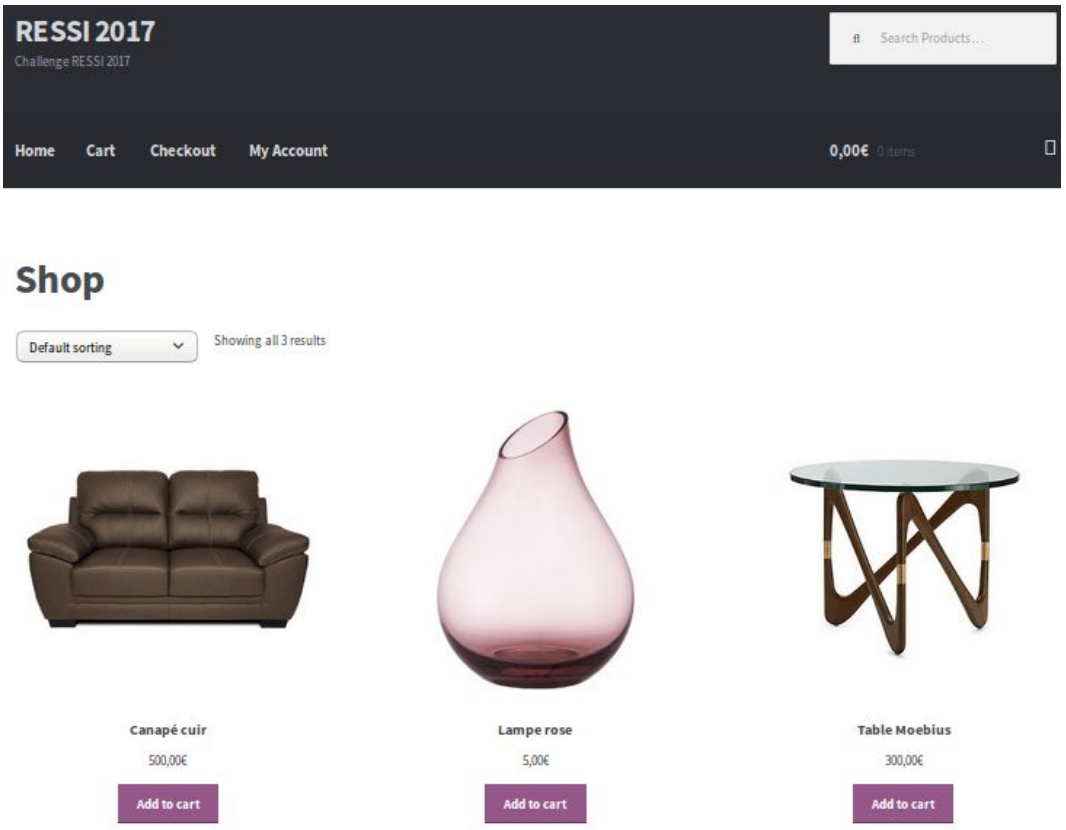


B - La préparation

I. Phase 1 : La plateforme de e-commerce

Premièrement, nous avons dû élaborer une application de e-commerce, qui correspond dans notre scénario à l'application du client. L'objectif final du challenge étant, pour rappel, de corriger les diverses vulnérabilités présentes sur un serveur web, tant au niveau du serveur en lui-même que de l'application derrière. Une application de e-commerce était particulièrement adaptée à nos besoins, car il est facile d'attribuer un score aux candidats : il suffit d'essayer régulièrement d'acheter sur le site, et si cela fonctionne, de relever le prix de l'achat et de l'ajouter au score.

Nous avons donc dédié la première semaine de notre projet à la création de cette plateforme. Nous avons pour ce faire utilisé **Wordpress**, tant pour sa simplicité, la rapidité avec laquelle on peut obtenir un résultat correct (car créer le site n'était pas le but du challenge, loin de la, nous devions donc y consacrer un temps restreint), et surtout pour ses divers plugins, notamment **WooCommerce**, permettant de transformer le Wordpress en une application de e-commerce très simplement. Par dessus tout, un des attraits de Wordpress, que nous n'avons malheureusement pas eu l'occasion d'exploiter, consiste en ses nombreuses vulnérabilités, ce qui allait bien dans l'esprit du challenge



.

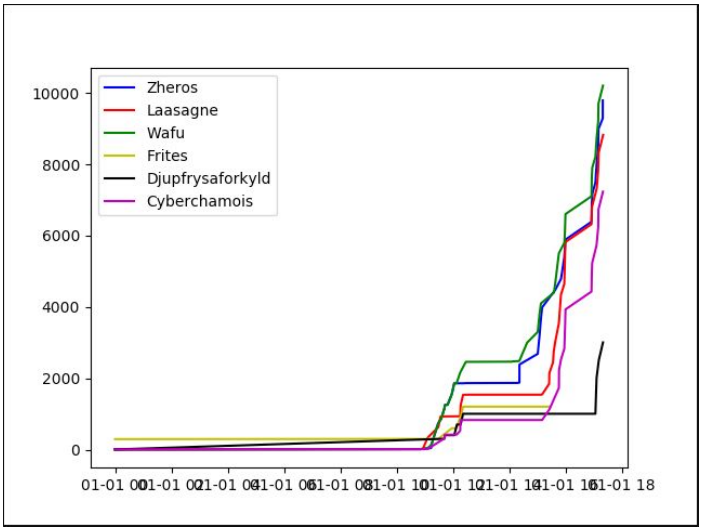
L'application en elle-même n'a pris que quelques jours à créer (configurer le site, créer des comptes, créer des produits, mettre des images, mettre des descriptions, mettre des commentaires …). Cependant, une fois notre application fonctionnant en local (grâce à WAMP, une plateforme de développement web Windows, nous devions l'exporter sur une machine virtuelle **Ubuntu Server 16.04 contenant LAMP**. Cela ne s'est pas fait sans encombre (notamment pour des raisons d'exportation de bases de données qui posaient problème, ou encore à cause de divers problèmes d'URL avec Wordpress …), mais nous sommes malgré tout parvenus à finir cela dans le temps qui nous avait été imparti par notre encadrant, à savoir une semaine.

II. Phase 2 : Le scoring

Maintenant, une fois cette machine créée, il fallait trouver une solution pour automatiser l'achat. Dans la mesure où il y avait 6 équipes, et que la procédure d'achat prenait en moyenne, lorsqu'elle était effectuée manuellement, une minute, cela aurait donné lieu à des manipulations fastidieuses le jour du challenge et, pire, aurait pu nuire à l'équité (oubli d'une équipe, différentiel de temps dans l'actualisation des scores des diverses équipes ....). Nous avons opté pour **Selenium IDE**, un plugin Firefox permettant d'enregistrer des listes d'actions. Ainsi, grâce à un simple clic, nous pouvions acheter les trois articles proposés en une dizaine de secondes, et ce en parallèle pour chaque équipe.

Cependant, une fois ces produits achetés, nous avions également besoin de récupérer le prix des produits, afin de calculer le chiffre d'affaire généré par le site de chaque équipe. C'est moins simple qu'il n'y paraît au premier abord, et nous avons dû télécharger un plugin supplémentaire, permettant de récupérer les logs générés par nos scripts. Il suffisait ensuite de récupérer la donnée du prix (en faisant une recherche par balise html, le prix étant dans la balise), de l'afficher dans le log avec un echo, d'enregistrer ce log quelque part, et enfin de le récupérer avec un script (Python).

Enfin, une fois les scores totaux récupérés, nous devions trouver un moyen de les faire connaître aux équipes. Une manière simple de le faire aurait pu être de créer un simple tableau avec les équipes et leurs scores, qui s'actualiserait en direct. Cependant, nous avons choisi une option qui montrerait de manière plus évidente aux candidats la présence d'un problème. En effet, il était peu probable qu'ils retiennent leur résultat d'une fois sur l'autre, donc ils n'auraient pas forcément relevé le fait qu'il ne vendent plus depuis plusieurs minutes. Nous avons donc présenté le scoreboard sous forme d'un graphique traçant le score en fonction du temps, pour chaque équipe. Une fois ce graphique créé (en Python) en récupérant à le score et l'heure par le biais des logs, il fallait maintenant pouvoir l'héberger sur un serveur HTTP. Nous en avons donc créé un (toujours en Python), de manière à pouvoir accéder à ce scoreboard grâce à l'URL localhost:80/scoreboard.py.



III. Phase 3 : Les vulnérabilités

En parallèle du scoreboard, nous avons commencé à implémenter diverses vulnérabilités, tant au niveau système qu'au niveau applicatif. Nous avons commencé par des failles extrêmement basiques, en créant des comptes administrateurs Wordpress ou sudoers sur la machine avec des mots de passe faibles. Les règles du challenge stipulaient que nous n'avions pas le droit de nous connecter sur le compte sudoer principal, utilisé par les candidats, afin de lancer des attaques, mais cela ne s'appliquait bien entendu pas aux autres comptes.

Type	Intitulé de l'exploit	Impact
Application	Second compte admin	Disponibilité des produits
Application	Shop for FREE!	Gratuité des produits
Système	Compte sudoers	Arrêt d'apache/iptables
Système	Backdoor simple	Reboot / Réactivation des exploits
Système	Backdoor chronée	Reboot / Réactivation des exploits
Système	Programme SUID root	Arrêt d'iptables
Système	Compte SSH secret	Réactivation des exploits
Rèseau	DoS HTTP	Disponibilité du site

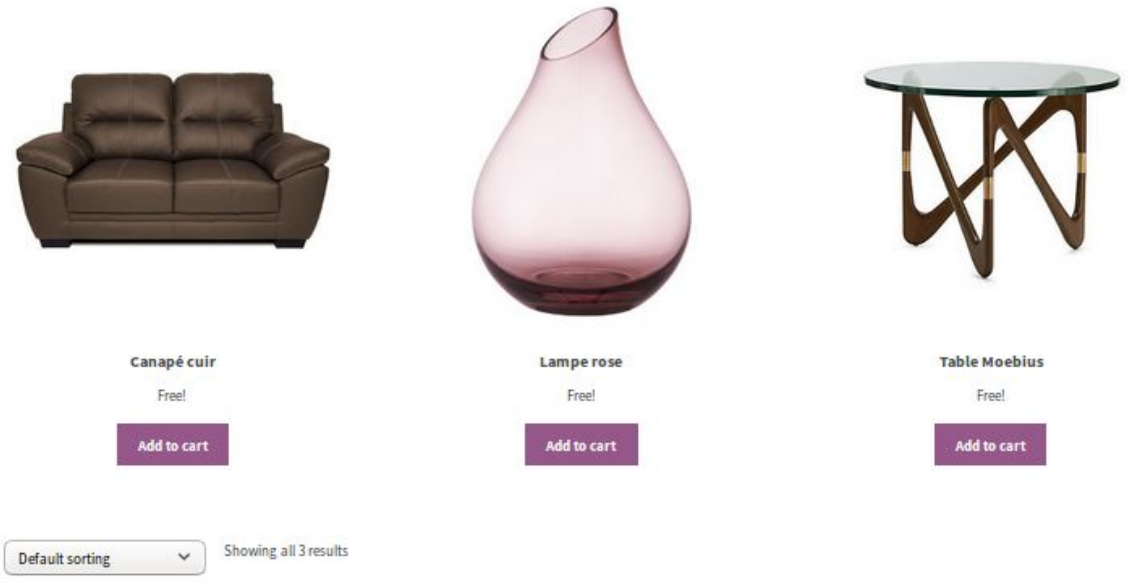
Ensuite, nous avons commencé à écrire des **backdoor** (toujours en Python), l'une dont le serveur était présent sur la machine virtuelle, ce qui était assez facile à détecter en regardant les ports ouverts, et une autre dont le client était sur la machine virtuelle, ce qui la rendait plus difficilement détectable. Nous avons ensuite demandé à un membre du comité d'organisation avec lequel nous travaillions, M. Nicomette, de transcrire en C ces backdoor (pour des soucis de discrétion : une fois compilé en binaire, le processus une fois lancé apparaîtra uniquement par son path dans la liste des processus, par exemple /usr/sbin/freeradius, tandis que s'il est exécuté en Python, il sera précédé de "python", ce qui est très suspect). De plus, nous avons modifié les **crontab**, à savoir, une liste de processus s'exécutant à heures données, de manière à y inclure une des backdoor. Elle s'ouvrait périodiquement toutes les quinze minutes, et se fermait deux minutes après, ce qui la rendait assez difficile à trouver.

La deuxième grosse vulnérabilité ayant été implémentée nous a été fournie par M. Nicomette. Elle consistait en un **faux serveur SSH**, se comportant en tout point comme un vrai, au détail près qu'il comportait un mot de passe et un nom d'utilisateur codés en dur dans les fichiers sources, qui permettait de prendre l'identité du compte principal sur lequel les participants travaillaient (nous n'étions pas autorisés à nous connecter dessus directement, mais cette faille se contentait de prendre son identité). L'ajout de la faille en lui-même n'a pas posé de souci particulier, la principale difficulté ayant été de trouver une version d'OpenSSH compatible avec les fichiers qui nous avaient été fournis.

La troisième, et principale vulnérabilité nous a également été fournie par M. Nicomette. Elle consistait en un service se lançant au démarrage de la VM, qui de manière périodique **arrêtait le serveur Apache** toutes les cinq minutes, **ajoutait le port 80 au firewall** toutes les cinq minutes et **lançait une backdoor** (et la relançait à chaque fois qu'elle était tuée). Contrairement aux autres vulnérabilités, celle-ci était critique : le site web était inaccessible tant que le service tournait (en réalité, il était aussi possible de relancer le serveur Apache et de vider le firewall toutes les cinq minutes, mais c'était assez fastidieux pour les participants). Ce service était nommé **agety**, de manière à ressembler à un autre service légitime, nommé agetty, qui sert à gérer plusieurs terminaux. Après quelques problèmes, liés à l'utilisation de caractères étranges, nous sommes parvenus à le faire fonctionner sur notre machine.

Une autre vulnérabilité, plus simple et également créée par M. Nicomette consistait en un **faux ls**. Ce dernier, une fois appelé 9 fois, se contentait de **nettoyer les iptables**, ce qui permettait d'effacer discrètement le firewalling réalisé par les candidats. A cause de conflits de versions (la machine sur laquelle le script avait été développé étant une Ubuntu Server 14.04 et la nôtre une 16.04) et de corruption de fichiers (les fichiers ayant été téléchargés sous Windows ce qui avait rajouté quelques caractères n'existant pas sous Linux), nous avons pris un certain temps à ajouter cette vulnérabilité, mais nous y sommes finalement parvenus une semaine et demi avant la date limite.

Enfin, une dernière vulnérabilité résidait dans la **configuration du Wordpress**. En effet, nous avions codé en dur une option faisant en sorte que si l'utilisateur était connecté mais n'était pas l'admin principal (le compte Wordpress auquel les candidats avaient accès), le prix du produit est réduit à 0. Ainsi, les candidats ne pouvaient pas voir que les produits se vendaient gratuitement avec leur compte, ou s'ils n'étaient pas authentifiés. L'intérêt du scoreboard sous forme de courbe était ici prédominant : le site des candidats fonctionnait parfaitement, mais leur score ne bougeait pas, ce qui leur permettait de relever la présence d'un problème.



IV. Phase 4 : La configuration réseau et le bêta-test

Une fois tout ceci fait, il nous fallait configurer le réseau local sur lequel le challenge allait se faire. Cette phase a majoritairement été réalisée par notre encadrant, M. Blanc, mais nous avons dû faire les tests qui y étaient liés, et adapter nos divers scripts, les backdoor notamment qui dépendaient de l'IP. Le réseau était dissocié en deux sous réseau, un sous-réseau contenant les acheteurs, les attaquants et le serveur HTTP sur lequel se trouvait le scoreboard, et un sous-réseau sur lequel se trouvaient les machines des candidats. Les machines des candidats ne pouvaient pas communiquer entre elles, afin d'éviter les attaques entre participants (ce n'était pas le but). Il fallait également rediriger les requêtes HTTP émanant du sous-réseau des candidats vers la machine contenant le scoreboard, à la fois pour qu'ils puissent voir leurs scores, et pour leur bloquer l'accès à l'interface d'administration du routeur.

Nous avons tout d'abord réalisé nos tests avec une borne WiFi Pineapple. Ils étaient concluants (bien qu'il y ait une forte latence quelque peu déplaisante …), mais dans la mesure où ce n'était pas l'appareil que nous allions utiliser le jour du challenge, nous les avons refaits avec la vraie borne, deux heures avant le bêta-test.

Ce dernier a permis de relever beaucoup de soucis mineurs (divers reliquats ...), mais également quelques soucis majeurs (les backdoor qui ne fonctionnaient plus, la présence de l'historique des commandes, le scoreboard qui ne fonctionnait pas sous Linux). Nous avons donc corrigé les backdoor, fait les arrangements nécessaires au niveau du scoreboard (transfert par clé des logs sur un PC disposant de Windows) et (supposément) nettoyé l'historique.

C - L'animation du challenge

Le challenge se déroulait à **Autrans**, près de Grenoble. Il faisait partie de la conférence annuelle de sécurité informatique **RESSI**, et s'étendait sur une journée parmi les trois de la conférence. Les candidats avaient reçu la VM la veille au soir, et avaient donc eu la soirée pour travailler dessus, mais seul trois groupes l'ont fait de manière intensive. Le challenge a commencé à 10h30 par une phase de configuration (nous nous sommes assurés que les candidats puissent bien se connecter au réseau), et les premières attaques ont commencé à 11h. Nous avons commencé par des attaques simples, exploitant au maximum nos backdoor avant que les candidats ne les trouvent. La plupart des équipes sont restées coincées un moment, car elles n'avaient pas profité de la veille pour corriger la vulnérabilité critique agety.

L'épreuve en elle-même s'est faite sans encombres, malgré quelques incidents (certains candidats avaient été déconnectés du WiFi et ne pouvaient pas s'y reconnecter, l'historique des commandes était toujours présent sur les machines à cause d'une fausse manipulation, un groupe a triché en bloquant leur compte principal en SSH, ce qui n'était pas autorisé pour des raisons d'administration (et l'attaque du faux serveur SSH reposait dessus). Nous avons donné ponctuellement des indices aux candidats, lorsque nous sentions que la frustration commençait à s'installer (leur site fonctionnant, mais leurs scores ne bougeant pas). La dernière phase de l'épreuve, de 16h30 à 17h30 était plus calme, la plupart des vulnérabilités ayant été trouvées par les candidats et leurs sites étant bien protégés (quelques candidats en ont d'ailleurs profité pour "enjoliver" l'interface de leur site).

Au final, l'épreuve a semble avoir été perçue comme un succès, tant de la part des candidats, que de la part du comité d'organisation. Les participants étaient curieux vis-à-vis des problèmes qu'ils avaient eu et sont venus nous poser des questions, tandis que d'autres nous provoquaient à chaque nouvelle vulnérabilité trouvée (avant d'assister, impuissants, à l'extinction de leur serveur à cause d'une autre vulnérabilité).

Annexes

Fiche technique

Les scripts d'achat

Création de différents profils Firefox

Configuration
OS utilisé : Linux
Navigateur Web utilisé : Firefox

Dans un terminal, tapez "firefox -no-remote -P&"
La procédure pour créer un nouveau profil est alors très bien indiqué.

Note importante : pour ouvrir plusieurs profils firefox en même temps, faire autant de fois la manipulation "firefox -no-remote -P&" (sans oublier le "&" !) que de profils que vous voulez ouvrir.

(Sous Windows, il y a ce tutoriel :
https://support.mozilla.org/fr/kb/utiliser-gestionnaire-profils-creer-supprimer-profils)

Pourquoi a-t-on créé plusieurs profils?
→ On ne peut utiliser qu'un seul script Sélénium à la fois par profil.

On aurait pu écrire un script qui parcourt les différents sites e-commerce des participants, mais :

- ça aurait été lourd, surtout si on ajoute ou on enlève au dernier moment des équipes.
- cela aurait avantagé ou désavantagé les équipes puisqu'il peut y avoir un gros décalage entre le premier site du script et le dernier.

Un script Sélénium ne concerne qu'un site à la fois. Nous avons donc copié ce script autant de fois qu'il y a d'équipes, et nous avons changé l'URL selon l'équipe.
Au final, pour chaque équipe était créé un profil Firefox avec sa copie des scripts Sélénium.

Remarque potentiellement utile : pour lancer rapidement les différents scripts, nous avions créé 9 espaces de travail sur notre Ubuntu. Dans chaque espace était ouvert un profil Firefox différent ainsi que son Sélénium. Le bouton pour lancer le script Sélénium était toujours au même endroit sur le bureau. Du coup : CTRL + ALT + flèches directionnelles puis cliquer, etc.

Les plugins Firefox nécessaires pour nos scripts Sélénium

- **Selenium IDE**

puis :

- **File Logging (Selenium IDE)** : pour récupérer les log des scripts Sélénium
- **Selenium IDE - SelBlocks** : pour pouvoir faire des conditions (if)

Pour voir le nom du profil :

-

Exemple de scripts d'achat

- exemple d'achat simple

- exemple d'achat avec réduction

Les vulnérabilités

Rappel :

Type	Intitulé de l'exploit	Impact
Application	Second compte admin	Disponibilité des produits
Application	Shop for FREE!	Gratuité des produits
Système	Compte sudoers	Arrêt d'apache/iptables
Système	Backdoor simple	Reboot / Réactivation des exploits
Système	Backdoor chronée	Reboot / Réactivation des exploits
Système	Programme SUID root	Arrêt d'iptables
Système	Compte SSH secret	Réactivation des exploits
Rèseau	DoS HTTP	Disponibilité du site

Détaillons un peu :

Vulnérabilités - Application :

- **Le second compte admin**

Sûrement la vulnérabilité la plus simple à réaliser : dans les configurations du WordPress, nous avons créé plusieurs comptes utilisateurs et admins. En particulier, le compte admin “thomas” avait “faible” comme mot de passe (ce qui est donc un mot de passe faible !).

Pour exploiter cette faille (ce manque de rigueur de la part de “thomas”), nous avons créé des scripts Sélénium permettant de se connecter en tant que “thomas”, puis d’ajouter des coupons de réductions ou encore de permettre à des produits d’être vendus gratuitement.

- **Shop for FREE!**

Vulnérabilités - Système :

- **Compte sudoers**
- **Backdoor simple**
- **Backdoor chronée**
- **Programme SUID root**
- **Compte SSH secret**

Vulnérabilités - Réseau :

- **DoS HTTP**

Le scoreboard

Liens divers

Site de RESSI :
https://ressi2017.sciencesconf.org/

Les plateformes de challenges citées :
https://www.newbiecontest.org/
https://www.root-me.org/
http://overthewire.org/wargames/

Nos travaux :
- Ancien drive : https://drive.google.com/drive/folders/0B3LMkUOD2uVTZE5xODNaTWJleG8?usp=sharing
- Nouveau drive : https://drive.google.com/drive/folders/0B3LMkUOD2uVTRDU3ZlpGSzRiQ2M?usp=sharing