# Documentation du code du projet de modélisation de l'attaque de Solarwinds sur Cyberrange

Tanguy Boisset, Vangelis Hoareau, Thomas Girard

2021-2022

## Introduction

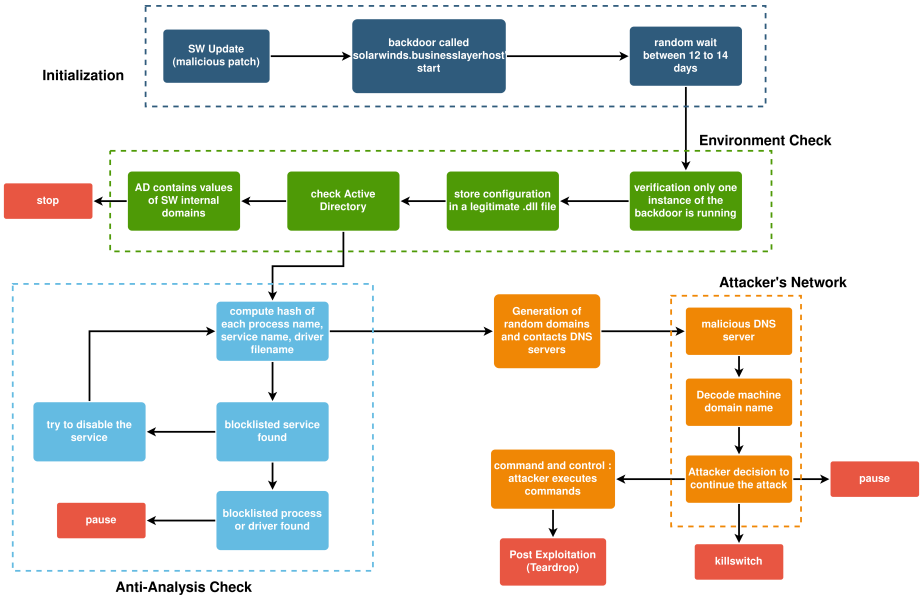


Figure 1: Description générale de Sunburst

Ce repository possède des dossiers pour chaque grande étape de l'attaque :

- 1-Initialization
- 2-EnvironmentCheck
- 3-Anti-AnalysisCheck
- 4-CommunicationC2Server

Les slides de la soutenance finale sont présente dans ce même dossier. Notre implémentation est uniquement en python, nous avons crées des .exe de ce code python (cf ci-dessous) avant d'exécuter le tout sur l'outil CyberRange d'Airbus.

Nous avons travaillé uniquement sur **Sunburst** et non sur la compromission initiale de SolarWind (cf slides de la soutenance finale).

## Description technique

### 1-Initialization

Ce dossier possède 2 sous-dossiers :

- Obfuscation : ce dossier contient les icônes, licences et fonctions permettants d'obfusquer au mieux l'exécution du malware afin de le faire passer pour un logiciel légitime.
- OrionUpdate : cette partie correspond à la mise à jour initiale de Orion avec la version compromise. Pour cela, le code recherche le fichier *OrionUpdateRequest.conf* qui contient l'IP d'un serveur distant puis télécharge le contenu et extrait les fichiers qui contiennent toutes les étapes suivantes du malware.

### 2-EnvironmentCheck

Ce dossier possède 2 sous-dossiers :

- CheckMalware : cela correspond à la vérification qu'une seule instance du malware est en exécution et que le fichier *SolarWinds.Orion.Core.BusinessLayer.dll.config* (qui sert à stocker des paramètres du malware) est bien présent. Un fichier *README.md* dans ce dossier apporte plus de précisions.
- DomainDiscovery : le malware vérifie que l'active directory présent ne contient pas certaines valeurs internes à Solarwinds afin d'être certain que le malware s'exécute chez un client de Solarwinds.

### 3-Anti-AnalysisCheck

Ce dossier contient la partie chargée de détecter si un processus ou un service représente une menace pour le bon déroulement de l'attaque. Le programme liste les processus et services en cours sur l'hôte, et possède une liste hardcodée de programmes que le malware tentera d'éviter en tentant de les arrêter ou en stoppant l'attaque.

### 4-CommunicationC2Server

Ce dossier possède 3 sous-dossiers :

- Backdoor : ce code permet la communication entre le malware et le serveur C2 de l'attaquant à l'aide de requête DNS puis de requêtes HTTP. Toute la partie de commuunication par requêtes DNS a été commenté car nous n'avons pas implémenté cela dans Cyberrange (nous avons hard-codé l'IP de l'attaquant sur Cyberrange). Le serveur de l'attaquant est un serveur Flask et ce dernier précise les commandes qu'ils souhaite exécuter à distance dans le fichier *commands.csv* puis récupère les informations dans le fichier *c2_log.txt*. Les slides 19 à 26 de la soutenance finale apportent davantage de précisions. Un fichier *README.md* est également présent dans ce dossier *Backdoor*.

- DGA : ce dossier correspond à l'algorithme de génération des noms de domaine. Le domaine généré permet la première communication avec l'attaquant avec des requêtes DNS. Ce code n'a pas été utilisé dans notre implémentation sur Cyberrange.
- MinimalBackdoor : ce code nous a permis de tester qu'une backdoor minimaliste était possible sur Cyberrange.

## Autre

### Compilation en .exe

Pour installer l'outil, lancer la commande : `pip install auto-py-to-exe`. cf : https://pypi.org/project/auto-py-to-exe/

Pour compiler en .exe, lancer `auto-py-to-exe` et indiquer les informations requises dans l'interface graphique. Le résultat apparait dans le dossier `output`.

Compiler le code sur une machine linux ne crée pas un exécutable windows, il faut être sur une machine windows pour cela.

Attention, si il y a des fichiers de configs (OrionUpdate), il ne pas oublier de le mettre dans la version compilée du malware !

### Upload d'un fichier sur Cyberrange :

Rendre exécutable le fichier : `chmod +x <fichier>`

Le placer dans un .tar : `tar -cvf <tar> <fichier`

## Bibliographie : principaux articles


- https://www.mandiant.com/resources/sunburst-additional-technical-details
- https://github.com/CyberSecOps/SolarWinds-Sunburst-Solorigate-Supernova-FireEye
- https://www.mandiant.com/resources/evasive-attacker-leverages-solarwinds-supply-chain-compromises-with-sunburst-backdoor
- https://blog.cloudflare.com/a-quirk-in-the-sunburst-dga-algorithm/
- https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/solarwinds-unique-dga