Livre blanc

Partie 3

Stratégies et Techniques pour échapper aux Antivirus modernes et contourner les EDR

NBS System
Groupe CELESTE

SOMMAIRE

1. PRINCIPE DE FONCTIONNEMENT DU HOOKING

Après avoir effectué des tests en utilisant les techniques d'injection mentionnées précédemment, il est apparu qu'elles étaient efficaces contre les antivirus mais pas contre les EDRs.

Nous avons donc dû approfondir notre compréhension du fonctionnement des EDRs en effectuant des recherches.

Après plusieurs investigations, nous avons compris que ces derniers utilisaient des mécanismes plus avancés que les antivirus, dont l'un des plus importants est le **Hooking**.

> Le Hooking est une technique utilisée par les EDRs pour surveiller les activités des processus en cours d'exécution sur un **Endpoint**. L'objectif principal du Hooking est de fournir une visibilité approfondie sur les actions effectuées par les programmes, ce qui permet de détecter les comportements malveillants et de répondre à ces menaces de manière **proactive**.

Voici le principe de fonctionnement du Hooking

1. PRINCIPE DE FONCTIONNEMENT DU HOOKING

Injection de code

Les EDR utilisent des techniques d'injection de code pour ajouter du code supplémentaire aux processus en cours d'exécution dans leur espace mémoire exécutable. Ce code supplémentaire est appelé "Hook" et permet de surveiller les activités du processus.

Surveillance des activités

Une fois le Hook injecté, l'EDR peut surveiller les activités du processus, telles que la création de fichiers, l'ouverture de connexions réseau, la modification des paramètres système, etc. L'EDR peut également surveiller les appels de fonctions effectués par le processus en question.

Analyse comportementale

Les activités surveillées sont ensuite analysées pour détecter les comportements malveillants. Les EDRs utilisent des algorithmes d'analyse comportementale pour détecter les activités suspectes, telles que la modification de fichiers système, l'installation de logiciels malveillants, etc.

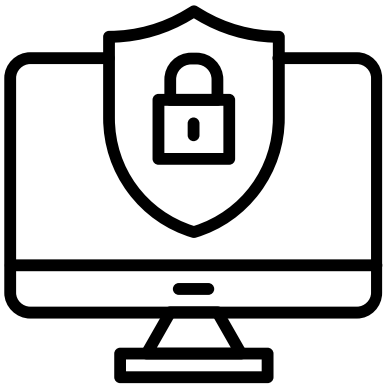
1. PRINCIPE DE FONCTIONNEMENT DU HOOKING

Déclenchement d'alertes

Si une activité suspecte est détectée, l'EDR génère une alerte pour informer les équipes de sécurité. Les alertes peuvent être déclenchées par des événements tels que la création d'un fichier malveillant, la modification des paramètres système, la tentative d'exécution d'une commande dangereuse, etc.

Réponse

Si une menace est confirmée, l'EDR déclenche une réponse automatique ou manuelle pour neutraliser la menace. Les réponses peuvent inclure l'isolation de l'Endpoint, la suppression du fichier malveillant, la fermeture des connexions réseau suspectes, la restauration du système à un état antérieur, etc.



1. PRINCIPE DE FONCTIONNEMENT DU HOOKING

L'intérêt du Hooking pour les EDRs réside dans le fait qu'il permet de **surveiller** les activités des processus en cours d'exécution de manière **fine** et **précise**. Cette surveillance approfondie permet de détecter les comportements malveillants qui pourraient autrement passer inaperçus pour la **BlueTeam**.

En outre, le Hooking peut être utilisé pour surveiller des processus spécifiques, ce qui permet de cibler des menaces potentielles et de détecter les comportements malveillants avant qu'ils ne causent des dommages.

Cependant, il convient de noter que le Hooking est une technique avancée qui nécessite une **connaissance approfondie** des systèmes d'exploitation et des processus en cours d'exécution. Une mauvaise utilisation du Hooking peut causer des **dysfonctionnements** sur les endpoints surveillés, ce qui peut entraîner des conséquences négatives sur la sécurité et la performance du système

2. TECHNIQUES DE HOOKING

Les EDRs utilisent plusieurs techniques de Hooking pour surveiller les activités des processus et détecter les menaces.

Dans notre projet, nous nous sommes concentrés spécifiquement sur les techniques de Hooking en **User land** plutôt qu'en **Kernel Land**, car elles sont les plus couramment utilisées par les EDRs pour plusieurs raisons :

Sécurité

Les **Hooks** en Kernel Land ont un accès privilégié au système d'exploitation et peuvent potentiellement causer des instabilités ou des plantages s'ils ne sont pas bien conçus ou bien implémentés. De plus, les Hooks en Kernel Land sont souvent la cible privilégiée des attaques de rootkits, qui tentent de masquer leur présence en modifiant le comportement du système d'exploitation.

Performance

Les Hooks en Kernel Land peuvent également impacter les performances du système en raison de la surcharge de traitement. En revanche, les Hooks en Userland ont moins d'impact sur les performances du système et sont plus faciles à déployer.

2. TECHNIQUES DE HOOKING

Flexibilité

Les Hooks en Userland sont plus flexibles que les Hooks en Kernel Land, car ils peuvent être placés sur des processus spécifiques plutôt que sur l'ensemble du système d'exploitation. Cette flexibilité permet aux EDR de surveiller les activités des processus spécifiques qui sont les plus susceptibles d'être ciblés par les attaques.

Cependant, il convient de noter que les Hooks en Userland ont également des limites, car ils ne peuvent surveiller que les activités des processus exécutés dans l'espace utilisateur. Les attaques qui visent directement le système d'exploitation et qui ne passent pas par l'espace utilisateur peuvent échapper à la surveillance des EDR utilisant uniquement des Hooks en Userland.

Ainsi, certains EDR peuvent utiliser une combinaison de Hooks en Userland et en Kernel Land pour offrir une surveillance plus complète et une protection plus efficace contre les menaces avancées.

Parmi les techniques de Hooking les plus utilisées en Userland par les EDRs nous trouvons

2. TECHNIQUES DE HOOKING

A. Principe de fonctionnement des antivirus

Cette technique consiste à remplacer les premières instructions d'une fonction existante par un code Hook qui intercepte les appels de fonctions. Cette technique est souvent utilisée pour surveiller les activités des processus en cours d'exécution. Voici un exemple de la technique :

« Inline Hooking » utilisée par un EDR : Supposons qu'un EDR souhaite surveiller les activités du processus malveillant **"evil.exe"** en interceptant l'appel à la fonction **CreateFileA**. Pour ce faire, l'EDR peut utiliser l'Inline Hooking pour remplacer les premières instructions de cette fonction par des instructions personnalisées qui redirigent l'exécution vers une fonction personnalisée.

1. Tout d'abord, l'EDR recherche l'emplacement de la fonction CreateFileA dans le code de "evil.exe".

2. Il modifie les premières instructions de cette fonction pour qu'elles redirigent l'exécution vers une fonction personnalisée, appelée par exemple **MyCreateFileA.**

2. TECHNIQUES DE HOOKING

3. La fonction MyCreateFileA est conçue pour enregistrer les informations sur la création de fichiers, telles que le nom du fichier, l'heure de création, l'emplacement, etc.

4. Lorsque "evil.exe" utilise la fonction CreateFileA, l'Inline Hooking redirige l'appel vers la fonction MyCreateFileA, qui enregistre les informations sur le fichier créé.

5. Après l'exécution de la fonction MyCreateFileA, l'EDR analyse les informations enregistrées pour détecter les comportements malveillants, tels que la création de fichiers suspects.

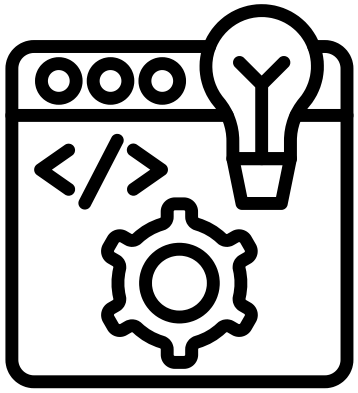
Il est important de noter que cette stratégie de Hooking peut être détectée et contournée par des **attaquants expérimentés**. La technologie des EDRs utilise donc souvent plusieurs techniques de Hooking Userland pour offrir une surveillance plus complète et une protection plus efficace du système d'exploitation contre les **menaces avancées**.

2. TECHNIQUES DE HOOKING

La technique actuelle présente un inconvénient car la **DLL** malveillante doit être déposée dans le système cible. Toutefois, dans la section suivante, nous examinerons une technique plus discrète qui permettra de charger la DLL ou n'importe quel **PE** sans le stocker sur le disque dur.

Cette méthode est plus complexe car elle nécessite une compréhension approfondie du mécanisme utilisé le loader de l'**OS** (parsing de l'**IAT**, **EAT**, etc.) pour charger les PE et plus particulièrement les DLLs en mémoire.

Grâce à cette compréhension, nous serons en mesure de développer notre propre loader de PE et d'intégrer la DLL directement dans le code source au lieu de la déposer dans le disque.



2. TECHNIQUES DE HOOKING

B. IAT Hooking

Le **IAT Hooking (Import Address Table Hooking)** est une autre technique de Hooking en Userland couramment utilisée par les EDRs pour intercepter les appels de fonctions dans les modules DLL chargés dans un processus. Elle consiste à modifier la table des adresses d'importation (IAT) d'un processus pour rediriger les appels de fonctions vers des fonctions personnalisées.

Cette technique permet à l'attaquant d'intercepter les appels de fonctions sans avoir besoin de modifier directement les instructions de la fonction. Il peut simplement modifier la table d'adresses d'importation pour rediriger les appels de fonctions vers une fonction personnalisée. Voici un exemple d'utilisation de la technique d'IAT Hooking par un EDR :

> Supposons qu'un EDR souhaite surveiller les appels à la fonction LoadLibraryA dans le processus "evil.exe". Il peut utiliser la technique d'IAT Hooking pour modifier l'adresse d'importation de cette fonction dans la table des adresses d'importation de "evil.exe" et rediriger les appels vers une fonction personnalisée.

2. TECHNIQUES DE HOOKING

1. L'EDR recherche l'emplacement de la table des adresses d'importation de "evil.exe" pour la DLL contenant la fonction LoadLibraryA.

2. Il modifie l'adresse d'importation de la fonction LoadLibraryA dans la table des adresses d'importation de "evil.exe" pour rediriger les appels vers une fonction personnalisée, appelée par exemple MyLoadLibraryA.

3. La fonction MyLoadLibraryA est conçue pour enregistrer les informations sur les DLL chargées par "evil.exe", telles que le nom de la DLL, l'emplacement, etc.

4. Lorsque "evil.exe" utilise la fonction LoadLibraryA, la table des adresses d'importation redirige l'appel vers la fonction MyLoadLibraryA, qui enregistre les informations sur la DLL chargée.

5. Après l'exécution de la fonction MyLoadLibraryA, l'EDR analyse les informations enregistrées pour détecter les comportements malveillants, tels que le chargement de DLL suspectes.

2. TECHNIQUES DE HOOKING

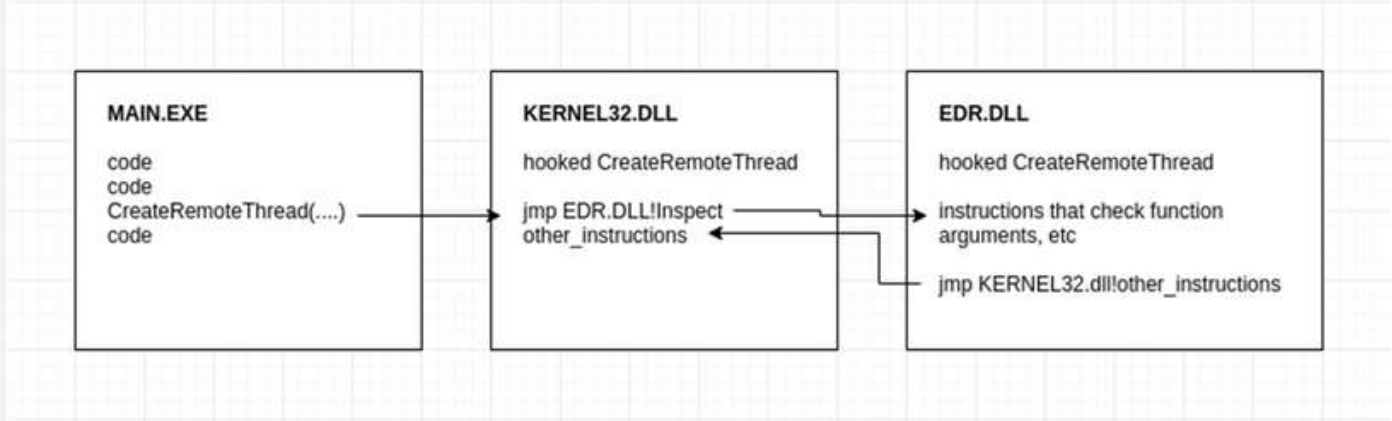


Figure 7 - Exemple d'InlineHooking utilisé par les EDRs

2. TECHNIQUES DE HOOKING

B. IAT Hooking

Le **IAT Hooking (Import Address Table Hooking)** est une autre technique de Hooking en Userland couramment utilisée par les EDRs pour intercepter les **appels de fonctions** dans les modules DLL chargés dans un processus. Elle consiste à modifier la table des adresses d'importation (IAT) d'un processus pour rediriger les appels de fonctions vers des fonctions personnalisées.

Cette technique permet à l'attaquant d'intercepter les appels de fonctions sans avoir besoin de modifier directement les instructions de la fonction. Il peut simplement modifier la **table d'adresses d'importation** pour rediriger les appels de fonctions vers une fonction personnalisée. Voici un exemple d'utilisation de la technique d'IAT Hooking par un EDR :

> Supposons qu'un EDR souhaite surveiller les appels à la fonction LoadLibraryA dans le processus "evil.exe". Il peut utiliser la technique d'IAT Hooking pour modifier l'adresse d'importation de cette fonction dans la table des adresses d'importation de "evil.exe" et rediriger les appels vers une fonction personnalisée.

2. TECHNIQUES DE HOOKING

1. L'EDR recherche l'emplacement de la table des adresses d'importation de "evil.exe" pour la DLL contenant la fonction **LoadLibraryA**.

2. Il modifie l'adresse d'importation de la fonction LoadLibraryA dans la table des adresses d'importation de "evil.exe" pour rediriger les appels vers une fonction personnalisée, appelée par exemple **MyLoadLibraryA**.

3. La fonction MyLoadLibraryA est conçue pour enregistrer les informations sur les DLL chargées par "evil.exe", telles que le nom de la DLL, l'emplacement, etc.

4. Lorsque "evil.exe" utilise la fonction LoadLibraryA, la table des adresses d'importation redirige l'appel vers la fonction MyLoadLibraryA, qui enregistre les informations sur la DLL chargée.

5. Après l'exécution de la fonction MyLoadLibraryA, l'EDR analyse les informations enregistrées pour détecter les comportements malveillants, tels que le chargement de DLL suspectes.

2. TECHNIQUES DE HOOKING

Figure 8 - Exemple d'IAT Hooking sur la fonction GetCurrentProcessId

3. EDR ET HOOKING SUR LA NTDLL

NTDLL (NT Dynamic Link Library) est une bibliothèque de liens dynamiques qui contient des fonctions importantes pour l'exécution de **Windows NT**. Les fonctions de la NTDLL sont utilisées par de nombreux processus et applications système de Windows, et sont donc une cible importante pour les attaquants qui souhaitent intercepter et modifier le comportement de ces processus.

Les EDRs utilisent souvent des techniques de Hooking pour surveiller les appels de fonctions de la NTDLL et détecter les comportements malveillants.

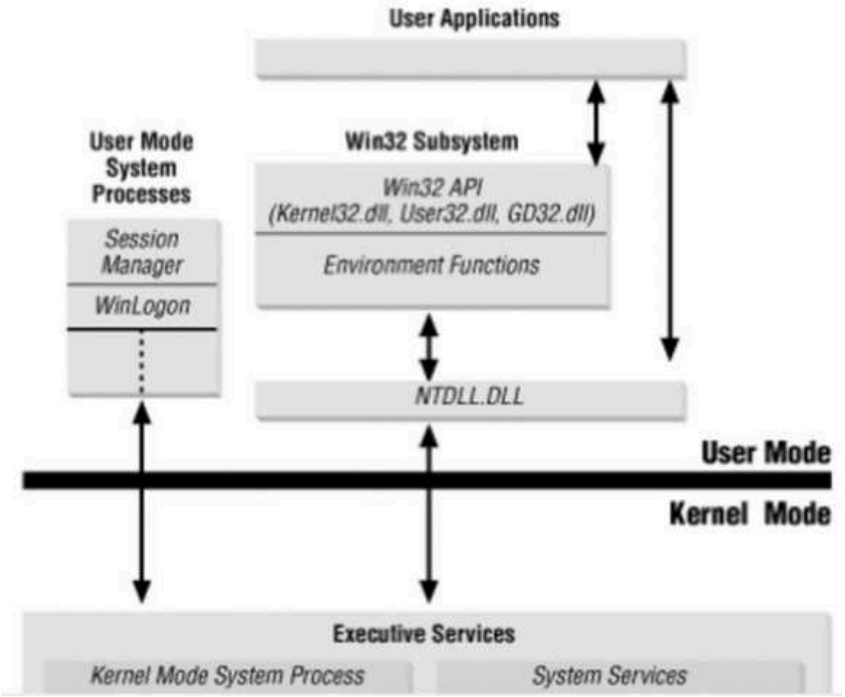


Figure 9 - NTDLL et l'architecture de l'OS Windows (Image credit : TechNet)

3. EDR ET HOOKING SUR LA NTDLL

Les appels aux fonctions de la NTDLL sont utilisés par de nombreux processus système et applications, et l'interception de ces appels peut fournir des informations précieuses sur l'activité d'un processus.

Les EDRs utilisent des techniques de Hooking pour intercepter les appels de fonctions de la NTDLL, telles que **NtCreateFile**, **NtOpenFile**, **NtReadFile**, **NtWriteFile**, **NtAllocateVirtualMemory**, **NtCreateSection**, **NtMapViewOfSection**, etc. Ils peuvent également utiliser différentes techniques de Hooking pour intercepter les appels de fonctions de la NTDLL.

Par exemple, l'Inline Hooking peut être utilisé pour modifier les instructions de la fonction de la NTDLL et rediriger les appels vers une fonction personnalisée, tandis que le IAT Hooking peut être utilisé pour modifier la table d'adresses d'importation de la NTDLL et rediriger les appels vers une fonction personnalisée.

3. EDR ET HOOKING SUR LA NTDLL

Les appels aux fonctions de la NTDLL sont utilisés par de nombreux processus système et applications, et l'interception de ces appels peut fournir des informations précieuses sur l'activité d'un processus.

Les EDRs utilisent des techniques de Hooking pour intercepter les appels de fonctions de la NTDLL, telles que NtCreateFile, NtOpenFile, NtReadFile, NtWriteFile, NtAllocateVirtualMemory, NtCreateSection, NtMapViewOfSection, etc. Ils peuvent également utiliser différentes techniques de Hooking pour intercepter les appels de fonctions de la NTDLL.

Par exemple, l'Inline Hooking peut être utilisé pour modifier les instructions de la fonction de la NTDLL et rediriger les appels vers une fonction personnalisée, tandis que le IAT Hooking peut être utilisé pour modifier la table d'adresses d'importation de la NTDLL et rediriger les appels vers une fonction personnalisée.

3. EDR ET HOOKING SUR LA NTDLL

En résumé, les EDR utilisent des techniques de Hooking pour intercepter les appels de fonctions de la NTDLL et surveiller l'activité des processus système et des applications.

L'interception des appels de fonctions de la NTDLL est une technique clé utilisée par les EDRs pour détecter les comportements malveillants, tels que la création de fichiers ou de processus suspects, le chargement de bibliothèques DLL malveillantes, etc.