Analyse de documents malveillants (partie 3) : documents Microsoft Office

Français

Après avoir vérifié comment configurer les machines virtuelles comme environnements sécurisés et présenté un processus d'introduction pour analyser les documents PDF suspects, nous sommes prêts à continuer avec les formats de fichiers Microsoft Office.

Problèmes de sécurité avec les documents Microsoft Office

En général, les documents Office ne sont pas dangereux s'ils contiennent uniquement les informations pour lesquelles ils ont été conçus : des pages contenant du texte et d'autres éléments imprimables pour MS Word, des cellules contenant des valeurs et des formules pour MS Excel, des diapositives contenant des éléments observables dans MS PowerPoint, etc.

Cependant, parmi les nombreuses fonctionnalités disponibles dans l'écosystème MS Office pour ajouter des fonctionnalités supplémentaires aux documents, il y en a une particulièrement intéressante du point de vue de la sécurité numérique : la possibilité d'intégrer des objets dans les documents. Les types d'objets que nous pouvons intégrer sont nombreux, comme la notation mathématique, les éléments multimédias, d'autres documents, etc. L'un de ces types est particulièrement puissant, car il permet l'exécution de code personnalisé, qui peut être utilisé pour nuire à l'utilisateur : les macros.

Macros

L'utilisation initiale des macros dans les documents MS consiste à exécuter facilement des tâches répétitives en les « enregistrant » une fois et en les « reproduisant » par la suite. Vous pouvez créer des macros sans avoir de connaissances en matière de programmation. Il suffit d'enregistrer les clics sur les boutons et les raccourcis clavier, et MS Office traduira ensuite l'enregistrement dans une série de commandes qui seront exécutées comme un « petit programme » actif à l'intérieur des documents.

Lorsque nous plongeons dans le fonctionnement des macros, nous constatons comment elles peuvent être utilisées comme armes et pourquoi elles sont si populaires dans les attaques d'hameçonnage pour infecter les ordinateurs par rapport aux autres techniques. Premièrement, les macros sont stockées sous forme de code écrit dans le langage de programmation Visual Basic for Applications (VBA), qui est bien documenté, simple à écrire et puissant. Visual Basic est aussi utilisé pour écrire des programmes autonomes, il a donc la capacité de faire des choses en dehors du document, comme télécharger et exécuter des fichiers, et modifier les paramètres du système par exemple. Les commandes associées à ces tâches sont également disponibles dans les macros MS Office, de sorte que nous pouvons écrire des macros qui tirent parti des commandes avancées disponibles et les utiliser pour

exécuter des tâches nuisibles, comme télécharger et exécuter des logiciels malveillants plus avancés, supprimer des fichiers, etc.

Deuxièmement, l'exécution de macros à partir de documents s'avère très facile. Les créateurs de documents peuvent configurer leur exécution automatique lors de l'ouverture du fichier, en cliquant sur un bouton, un lien ou tout autre élément donné, entre autres déclencheurs. Dans le cas des fichiers inconnus, MS Office nous avertira que leurs macros pourraient être dangereuses et les bloquera, mais il suffit souvent d'un ou deux clics pour désactiver cette protection et exécuter les macros malgré tout. Cette situation est très attrayante pour les acteurs malveillants, car ils peuvent utiliser des macros et nous convaincre qu'elles sont sûres par le biais d'arguments convaincants propres à chaque campagne d'hameçonnage.

En comparant les documents PDF aux documents MS Office contenant des macros concernant les activités malveillantes, les documents MS Office offrent plus de possibilités de commandes sur les appareils sur lesquels ils sont exécutés, ce qui les rend plus puissants et aussi plus populaires que les PDF. Cette flexibilité est également la raison pour laquelle nous nous concentrons sur l'utilisation de macros autorisées dans les documents malveillants par rapport à d'autres moyens d'utiliser des documents MS Office comme armes. Si vous souhaitez en savoir plus sur les différentes façons dont ces fichiers peuvent être utilisés pour attaquer les utilisateurs utilisant des versions obsolètes d'Office, nous fournissons des liens vers d'autres références à la fin de ce cours.

Vulnérabilités d'Office
Il existe de nombreuses façons documentées utilisées pour exploiter les macros ou les documents Office en général. Cependant, beaucoup de ces méthodes créatives ne sont pas exploitables dans les instances entièrement mises à jour de MS Office.

Comme tout autre programme, MS Office peut contenir des vulnérabilités connues ou inconnues qui peuvent permettre la compromission d'un appareil, sans même recourir aux macros.

Les formats de document MS Office « anciens » et « nouveaux »

Depuis 2003, Microsoft Office a modifié la façon dont les documents sont créés par défaut, y compris via de nouvelles extensions de fichier, de sorte que les nouveaux documents MS Word sont stockés avec l'extension « .docx » au lieu de « .doc » et ainsi de suite. Même lorsque la structure interne de ces deux formats de fichiers est différente, les macros sont stockées de manière similaire. Les indications fournies dans ce document s'appliquent donc aux anciens et aux nouveaux formats de fichier.

Si vous souhaitez en savoir plus sur les conventions de stockage des macros et de nombreux autres types d'objets, vous pouvez faire des recherches sur le lien et l'intégration d'objets (Object Linking and Embedding, ou « OLE »), qui est encore utilisé et adapté aux nouveaux formats de fichiers, y compris les documents MS Office.

Analyse des documents MS Office

Pour commencer à explorer les façons de détecter quand des macros sont incluses dans les documents MS Office, nous utiliserons oledump.py, un outil Python développé par Didier Stevens, le même auteur des outils proposés dans la partie précédente axée sur les fichiers PDF. Nous utiliserons également une série de fichiers d'exemples pour montrer comment les documents MS Office fonctionnent et comment les macros peuvent être détectées et examinées, également à partir des documents de formation de Didier Stevens.

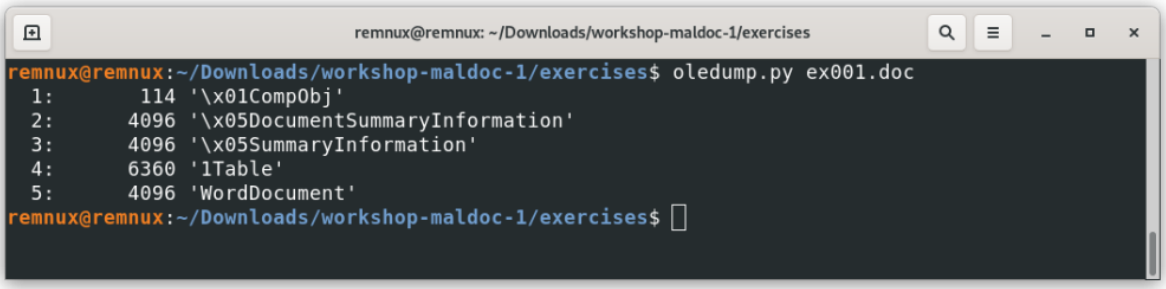
Le processus pour commencer l'analyse des documents MS Office est très similaire à celui que nous avons utilisé pour les fichiers PDF. Tout d'abord, nous énumérons les différents éléments présents dans le fichier, nous identifions les objets intéressants en termes de sécurité, puis nous tentons d'obtenir le contenu réel de ces éléments pour voir s'ils contiennent des menaces.

Oledump.py

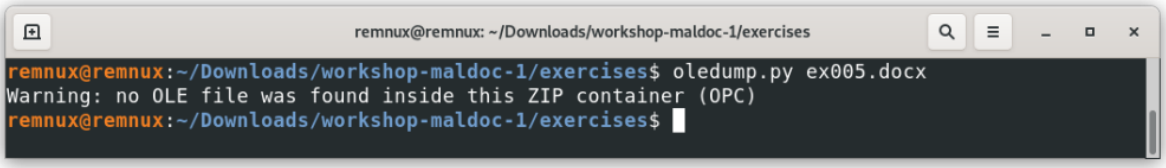
L'utilisation principale de cet outil est d'énumérer tous les objets OLE inclus dans un fichier spécifique et d'afficher leurs contenus. L'utilisation de base de l'outil est la suivante :

```
oledump.py ex001.doc
```

Lorsque ex001.doc est le nom du document que nous voulons analyser :

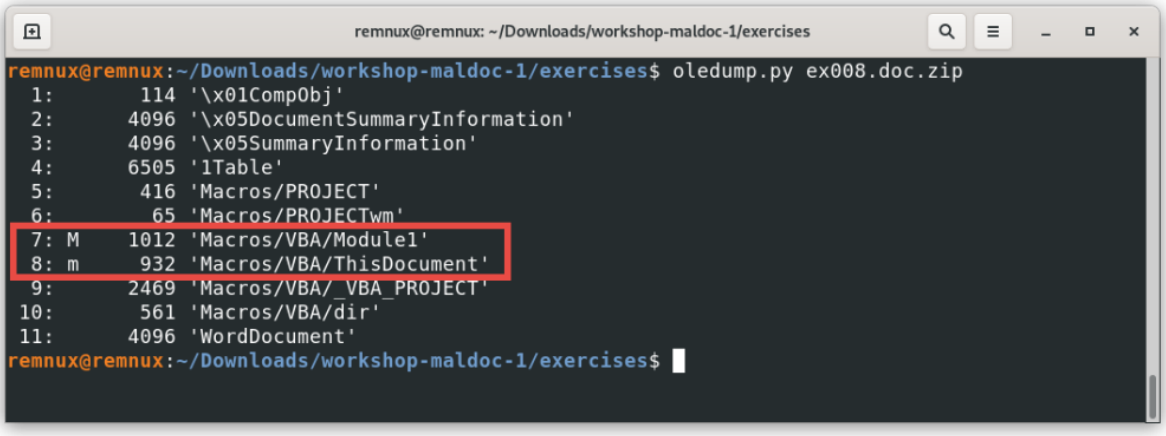


Nous pouvons voir ici tous les éléments d'un fichier sans macros ou autres objets insolites intégrés dans un « ancien » type de fichier MS Office. en faisant la même expérience pour un fichier avec le « nouveau » format (après MS Office 2003), nous obtiendrons quelque chose comme ceci :



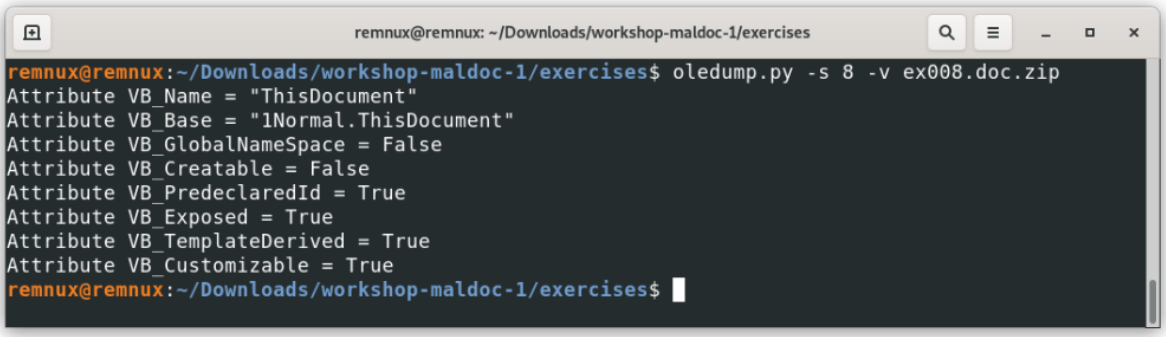
Nous pouvons voir ici que plus d'éléments dans le fichier .doc d'exemple sont stockés en tant qu'objets OLE, alors que dans le fichier .docx d'exemple, nous avons un document fonctionnel sans utiliser d'objets OLE. Cela est dû au fait que les documents .docx sont compressés sous forme de fichier .zip avec principalement des fichiers .xml à l'intérieur (vous pouvez même essayer de changer un fichier .docx|.xlsx|.pptx sécurisé en extension .zip et l'ouvrir), et n'utiliser que des données formatées OLE lorsque cela est nécessaire.

Lorsque nous ouvrons un fichier contenant des macros, le résultat inclura de nouveaux éléments :



Si nous regardons de près cet exemple, nous pouvons voir que le fichier que nous avons passé à la commande est un fichier .zip. Dans ce cas, c'est un fichier .zip contenant un document MS Word. De plus, le fichier .zip est protégé par un mot de passe avec le mot de passe « infecté ». C'est une pratique courante dans la communauté d'analyse de logiciels malveillants, et oledump.py le considère comme une entrée valide et gère toute la décompression, et passe ensuite le document pour l'analyser automatiquement pour nous.

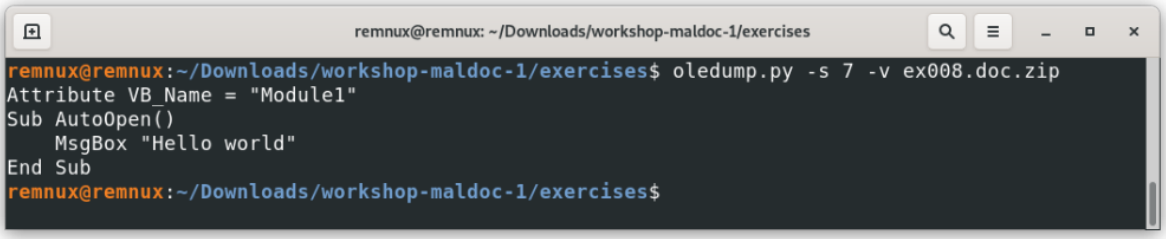
Dans ce résultat, nous voyons 2 objets différents qui sont identifiés par la lettre « m » ou « M ». Cela signifie que ces objets spécifiques contiennent des macros. Utilisons la commande `-s` dans oledump.py pour voir le contenu de ces flux, en commençant par l'objet (ou le flux 8).



Nous avons utilisé la commande `-s` pour sélectionner l'objet identifié par le chiffre 8, et la commande `-v` pour décompresser le contenu, car VBA peut compresser du code par défaut,

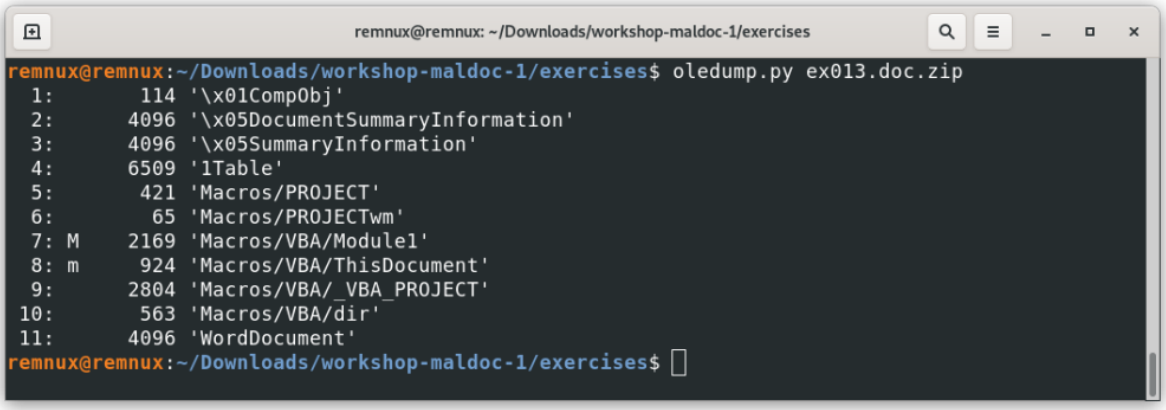
il est donc prudent d'inclure cette commande lors de la requête d'objets contenant des macros.

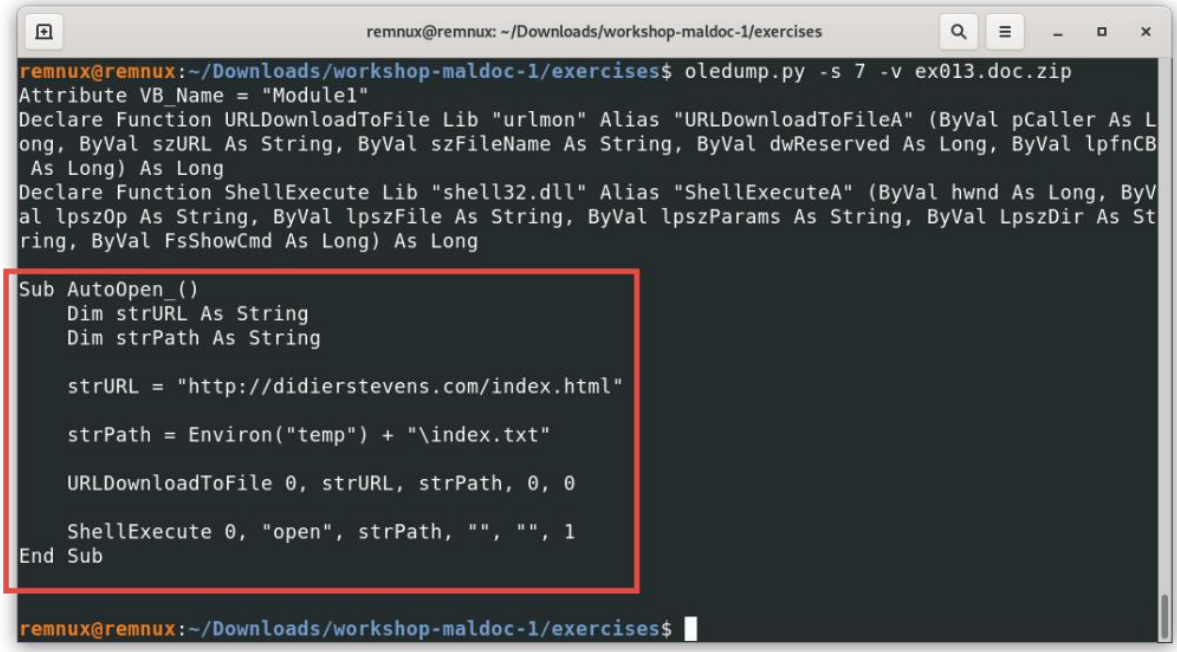
En examinant le contenu, nous voyons quelques déclarations d'attributs, celles-ci sont faites par défaut par VBA et ne sont même pas visibles pour le créateur du document. Ce code n'est donc pas considéré comme personnalisé ou nuisible dans notre examen. Le code connu comme étant inoffensif par l'outil est identifié avec un « m » en minuscules, en le supposant comme étant sûr et moins intéressant pour une analyse plus approfondie. Analysons l'autre objet contenant une macro.



Pour donner un peu de contexte, cette macro utilise la commande MsgBox qui lance une fenêtre de dialogue avec le message « Hello world » dans ce cas. De plus, AutoOpen() indique au programme que cette macro doit être exécutée automatiquement lors de l'ouverture du fichier. En pratique, un document inconnu essayant d'exécuter une macro AutoOpen() déclenchera une alerte de sécurité. Cependant, selon le contexte, l'utilisateur peut être trompé pour contourner l'avertissement et exécuter la macro malgré tout.

Pour explorer la façon dont cela peut être utilisé comme arme, examinons le fichier suivant :

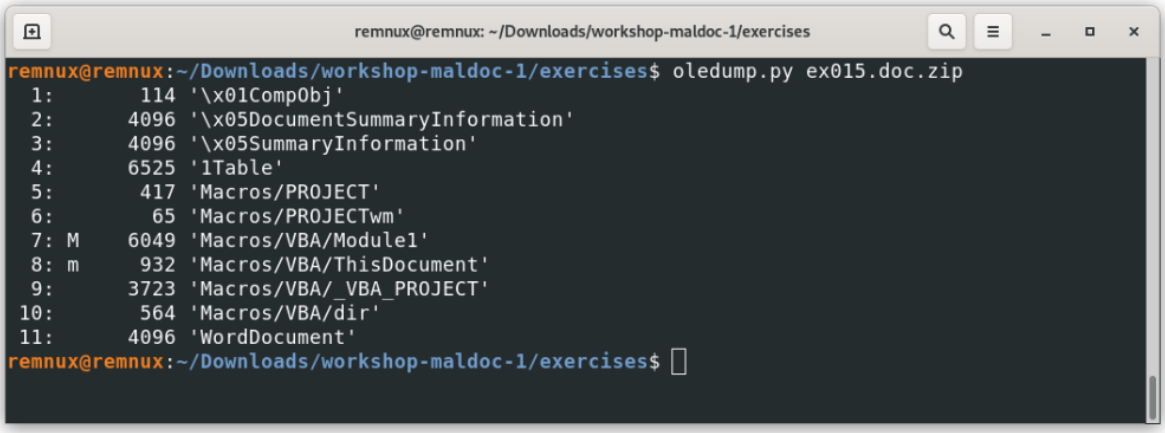


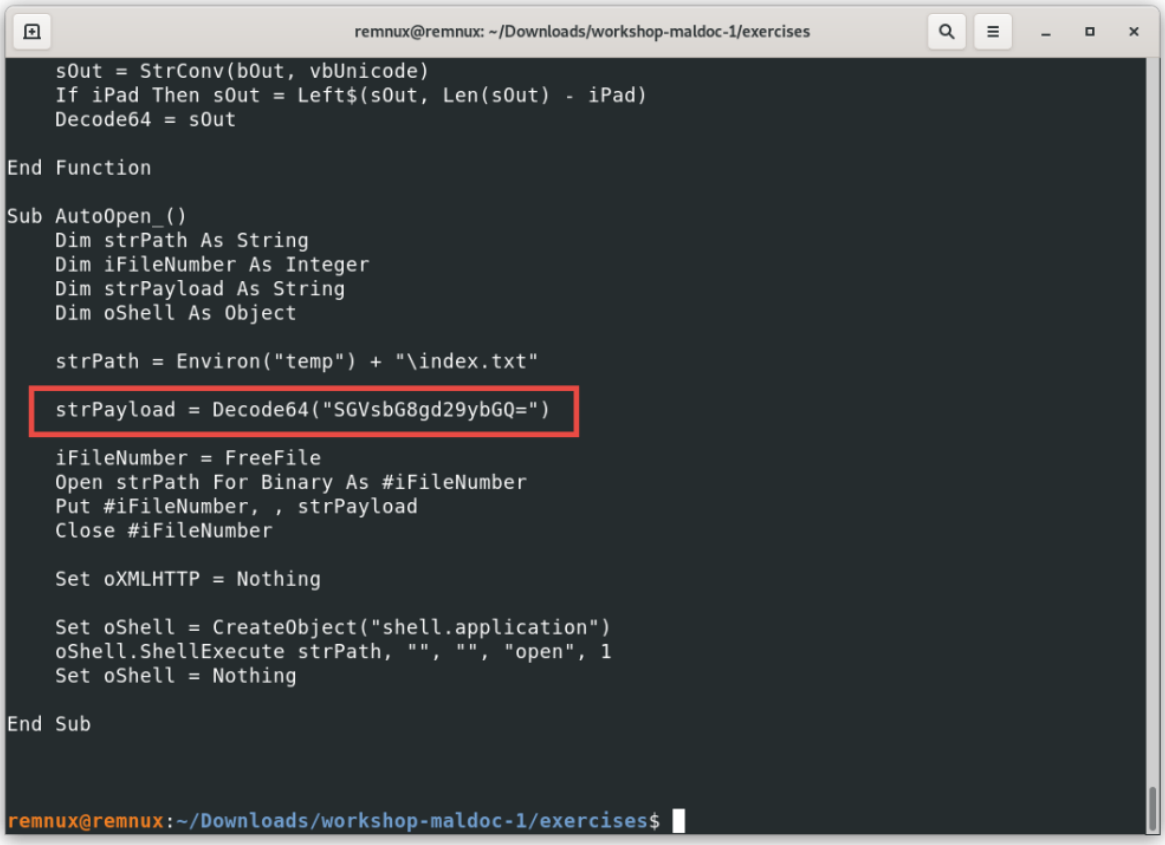


Ici, la macro qui nous intéresse est un peu plus complexe qu'une fenêtre de dialogue avec un message de texte. À nouveau, nous pouvons voir que la fonction AutoOpen() est utilisée pour exécuter le code ci-dessous lors de l'ouverture du document. En examinant le code, avec un peu d'aide pour vérifier les commandes utilisées, nous pouvons déduire que la macro essaie de télécharger le contenu d'une URL dans un fichier du répertoire temporaire de notre machine et d'exécuter le fichier téléchargé.

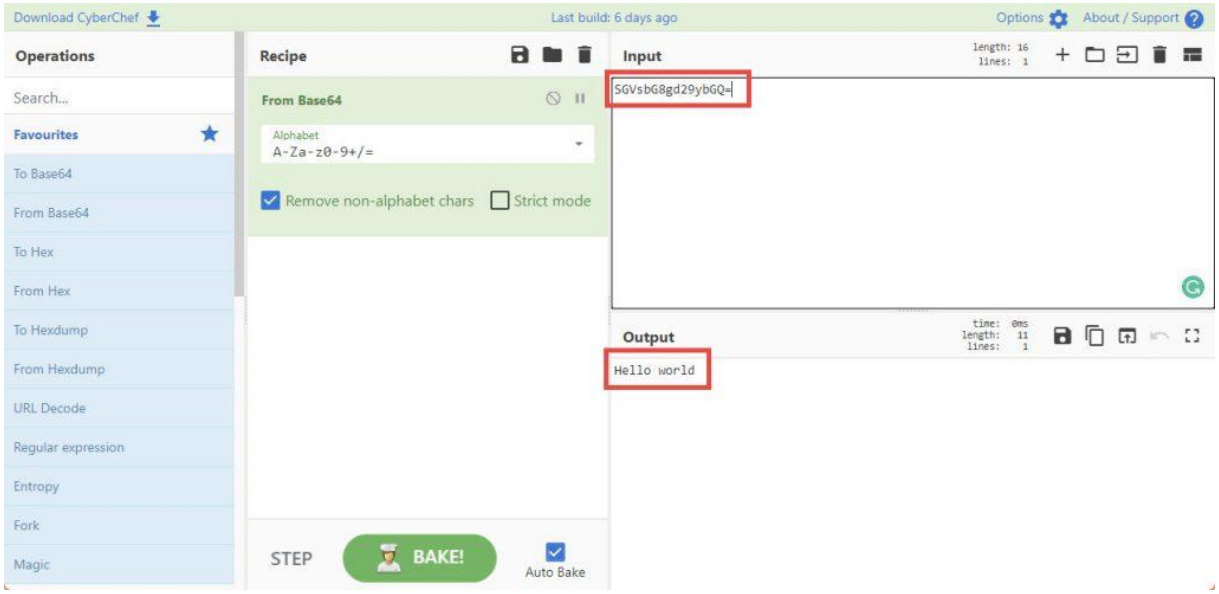
Dans ce cas, il semble que le fichier remplit un fichier texte qui devrait être inoffensif, mais avec la bonne URL et le bon type de fichier utilisé pour télécharger le contenu, une macro comme celle-ci permet d'écrire et d'exécuter des programmes ou autres artefacts nuisibles avec une interaction minimale de l'utilisateur. Il est également important de mentionner que cette macro est une version simplifiée des menaces plus réelles, qui obscurcissent généralement leur code pour éviter leur détection par les logiciels antivirus et peuvent exécuter des actions plus élaborées, comme ajouter le logiciel malveillant téléchargé aux programmes de démarrage ou aux tâches planifiées pour devenir persistant dans le temps, entre autres.

Souvent, l'analyse de macros plus complexes nécessitera des compétences qui ne sont pas couvertes dans ce document, comme déchiffrer du code et comprendre comment, grâce à l'utilisation de différentes commandes et structures de données, nous pouvons obtenir le code final exécuté afin de déterminer ce qu'il fait (désobscurcissement). Un exemple encore très simple montrant une technique commune pour brouiller le contenu peut être trouvé en consultant le fichier suivant.





Ici, au lieu d'utiliser du texte brut, le créateur de la macro a utilisé le schéma de codage base64 afin de compliquer la lecture des informations qu'il tente d'exécuter. Pour cet exemple, il existe de nombreux outils qui peuvent nous aider à décoder cette variable, l'un d'eux est CyberChef, une application Web dans laquelle nous pouvons saisir des données et exécuter certaines opérations pour obtenir un résultat. Dans ce cas, nous avons :

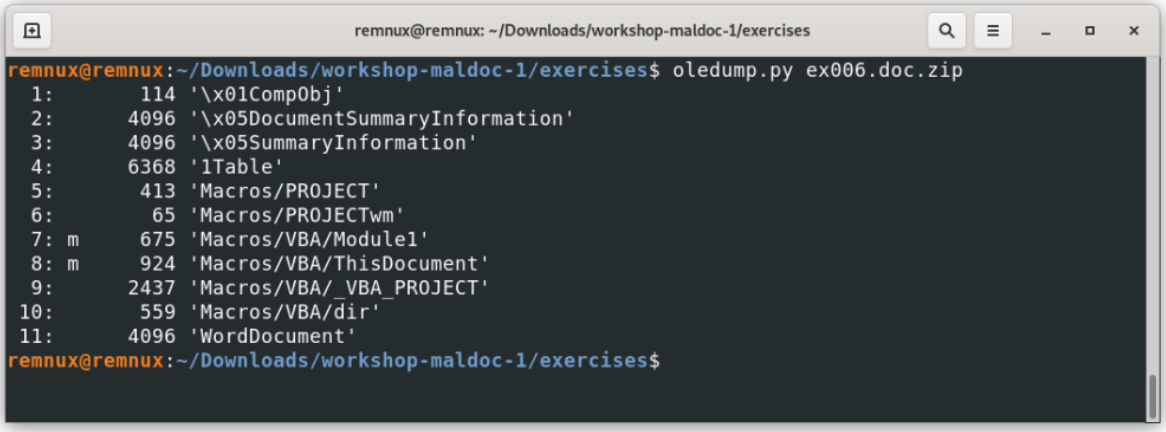


Avec ces exemples et références, nous devrions être en mesure de savoir si un fichier a des macros intégrées à l'aide de l'outil oledump.py, si un fichier devrait être analysé de façon plus approfondie, et dans le cas où son code est assez simple, ce que la macro tente de faire. Dans le cas des documents comprenant des macros très complexes, le conseil est de chercher de l'aide pour analyser le fichier plus en profondeur et ne jamais tenter d'exécuter le fichier dans nos environnements, car cela pourrait avoir des effets dévastateurs sur nos appareils s'ils s'avéraient infectés.

Maintenant que nous avons appris quelques processus de base pour analyser les fichiers PDF et MS Office, nous sommes prêts à examiner certaines stratégies défensives pour nous protéger contre les documents malveillants dans la partie finale suivante.

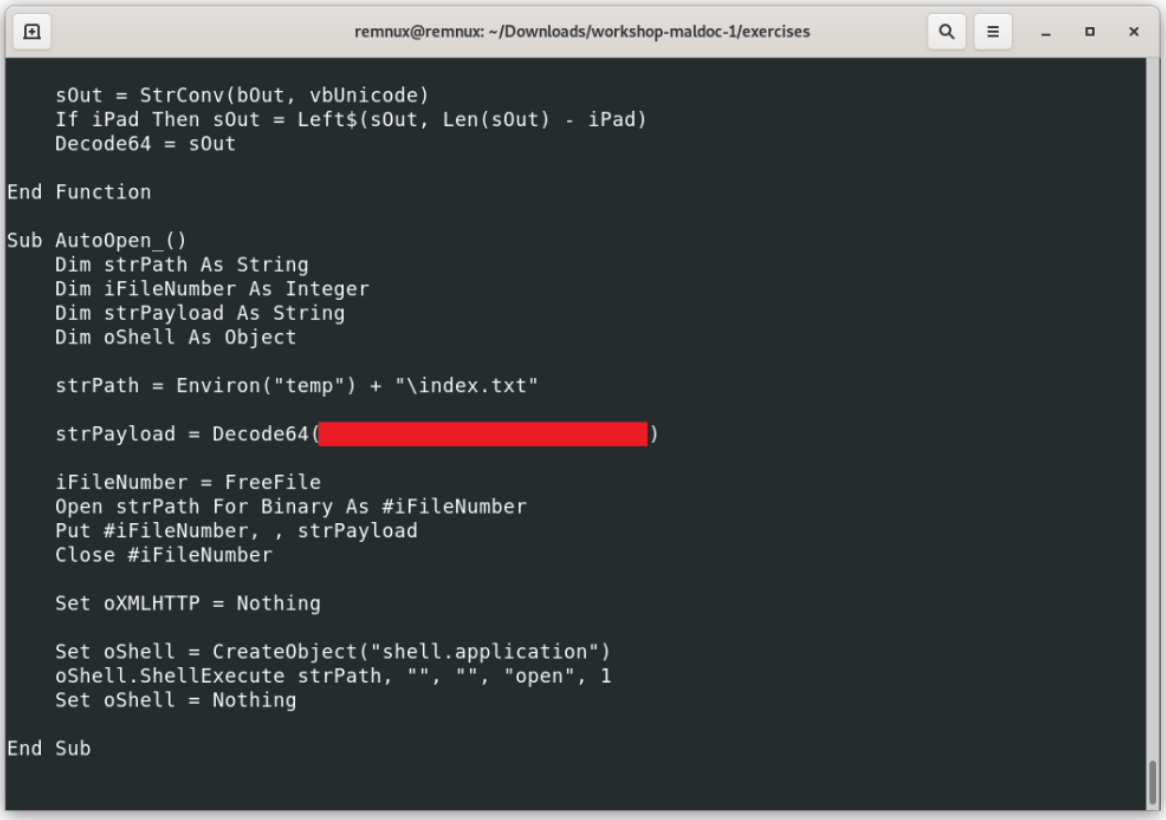
Défis

Question : en appliquant le même processus au fichier ex006.doc.zip, nous obtenons ce résultat. Laquelle des hypothèses suivantes est vérifiée avec les informations que nous avons obtenues jusqu'à présent ?



- 1. Le fichier n'a pas de macros
- 2. Le fichier a des macros personnalisées comme tous les autres exemples abordés
- 3. Le fichier contient des macros inoffensives, mais pas de macros personnalisées

Question : en appliquant le même processus au fichier ex016.doc.zip, nous constatons une macro similaire à la dernière décrite ci-dessus, où la macro décode quelque chose en base64. Qu'est-ce qui est passé à la fonction Decode64 ? (Indice : aaaaaaaaaaaaaa.aaaaaaa.aaaa)



- Je veux voir la réponse

Et ensuite ?

Maintenant que nous avons une meilleure idée de la façon dont les fichiers PDF et MS Office peuvent être évalués pour y détecter d'éventuels codes malveillants, nous pouvons mieux comprendre comment proposer des mesures défensives et nous pouvons examiner les conseils finaux sur la façon de mener ce genre d'évaluation initiale.

Bonus : informations complémentaires sur la sécurité de MS Office

- Oletools : un autre outil célèbre pour analyser les fichiers MS Office
- Liste des vulnérabilités connues de Microsoft Office (la plupart ne faisant pas appel à des macros)
- CVE-2022-30190 ou nom de code « Follina » : une vulnérabilité récente qui abuse des documents pour interagir avec les fonctionnalités de dépannage de Windows.
- « Sans compromis : décompresser une macro Excel malveillante », un cas intéressant d'exploration d'un fichier malveillant étape par étape.
- Aide-mémoire de l'analyse des documents malveillants, les conseils rapides de Lenny Zeltser pour analyser des documents suspects.