Cours d'introduction au reverse-engineering et mécanismes de sécurités des OS Windows
Évaluation

Alexandre Gazet
Julien Lenoir
Raphaël Rigo

TLS-SEC 2019/2020

Abstract. L'équipe de journalistes d'investigation du `Times of Toulouse` utilise un logiciel (propriétaire) de chiffrement de fichier afin d'échanger des documents avec ses sources: `newlocker.exe`.

Plusieurs fuites et/ou compromission ont entraîné une perte de confiance dans ce logiciel. Jeunes professionnels surentraînés, il vous est demandé d'évaluer le produit et de rendre un avis quant à son utilisation possible.

Souvenez-vous qu'il vous faudra regarder le code droit dans les yeux pour en percer les mystères.

Note importante: toutes vos réponses doivent être justifiées. Pensez par exemple à expliciter l'emploi des outils qui vous paraissent appropriés et à décrire votre démarche. L'analyse et les manipulations de haute précision peuvent-être réalisées dans la VM fournie durant le cours.

1 Objectifs de la mission

1.1 Préliminaires

- Quel est le *timestamp* contenu dans l'en-tête du binaire ?
- Combien de sections sont présentes dans le binaire ? Combien contiennent du code exécutable ? L'une d'elle vous paraît-elle suspecte?
- Proposez une analyse rapide des fonctions **importées** par le binaire. Peut-on en déduire quelque chose ?
- De la même manière, proposez une analyse rapide des fonctions **exportées** par le binaire.

1.2 Analyse fonctionnelle

Le binaire est fourni en l'état par un informateur travaillant pour le journal. Afin de vous aider dans la prise en main du produit, cette personne vous a fourni les fichiers suivants:

- "*sample.raw*": le fichier original en clair
- "*sample.raw.ycrypt*": le fichier chiffré
- "*sample.raw.yclear*": le fichier déchiffré
- "*note.txt*": fichier texte contenant les commandes utilisées (ainsi donc que le mot de passe)

1.3 Reverse-engineering

Posez les bases et analysez les fonctions:

- `0x40102A` et `0x40105D` en proposant un pseudo-code.
- Quelles sont toutes les options supportées par la ligne de commandes?
- Pour chacune d'entre elles indiquez le nombre d'arguments attendus ainsi que leurs sémantiques.
- La fonction `0x401354` semble être importante. Que vous permet-elle de découvrir sur la ligne de commande?

Tip: pour la suite, aidez-vous au maximum des informations présentes:

- De nombreux messages de debug/log semblent présents.
- Certains symboles (noms de fonctions) n'ont pas été retirés.
- Une fonction apparaissant en violet/rose dans `IDA` est une API exposée par le système, donc documentée. Recherchez les constantes pour bien comprendre la sémantique des appels.

1.4 Génération de mot de passe

Le logiciel propose une fonctionnalité de génération de mot de passe **forts**. Analysez l'algorithme et donnez un avis sur sa sécurité.

Pour ce genre d'analyse on s'attache à déterminer précisément quelles sont les sources d'entropie (d'aléa). L'algorithme est très linéaire n'hésitez pas à le ré-écrire à la main pas à pas dans un script.

1.5 Analyse du chiffrement

Le cœur du logiciel est la fonctionnalité de création d'archive chiffrée. La seule information dont nous disposons est que tous les algorithmes de chiffrement utilisés sont standards.

Pour vous aider, vous pourrez découper votre analyse en plusieurs étapes:

1. Dérivation d'une clé de chiffrement à partir d'un mot de passe.
2. Création d'une archive chiffrée. Des metadata sont stockées dans un en-tête:
   - Quel est le format de l'en-tête ?
   - Proposez une structure `C` le décrivant.
3. Chiffrement des données.

4. Comment est vérifiée l'intégrité des données déchiffrées?

Tip: pour chacune de ces étapes, il vous sera facile d'identifier une fonction correspondante à partir de messages de log.
Tip2: pour reconstruire une structure on s'intéresse souvent à initialisation. Taille de l'allocation initiale puis les différents offsets accédés. *cf.* "IDA: Structures Tutorial"[1].

2 Compromission

Après recoupement des traces forensic, toutes les fuites ayant impactées des employés du journal ont eu lieu suivant un mode opératoire commun: un journaliste reçoit une archive chiffrée avec un mot de passe valide.

Une fois déchiffrée, l'archive ne contient qu'un fichier bénin; toutefois un fort trafic sortant est rapidement observé depuis le poste de la victime. Se pourrait-il qu'une vulnérabilité soit exploitable dans le binaire ?

[1] https://hex-rays.com/products/ida/support/tutorials/structs.shtml

Quelques points clés à vérifier seront:

- Des crashs ont été reportés pour les archives de grandes tailles. Le logiciel prévoit-il une taille maximalle des archives?
- À l'aide du script PowerShell `Get-PESecurity` (vu durant le TP), est-ce qu'un mécanisme de mitigation bien connu est absent?
- Une section du binaire est quand même vraiment étrange:
  - Quelle est la taille des données utiles ? (Note: pour des raisons d'alignement mémoire, la taille des sections est souvent arrondie au 0x1000 supérieur et les données simplement padées avec des zeros.)
  - Sans analyser en détail son contenu, retrouvez-vous un élément déjà vu en TP ?

À l'aide de tous les éléments collectés durant votre analyse, proposez une explication de la vulnérabilité, accompagnée d'une preuve de concept déclenchant l'apparition d'un "*calc.exe*".

3 Votre opinion

Formulez une recommandation concernant l'utilisation de ce logiciel et la confiance que l'on peut y placer. Vous justifierez votre réponse à l'aide d'éléments techniques; en particulier si des points vous paraissent suspects.

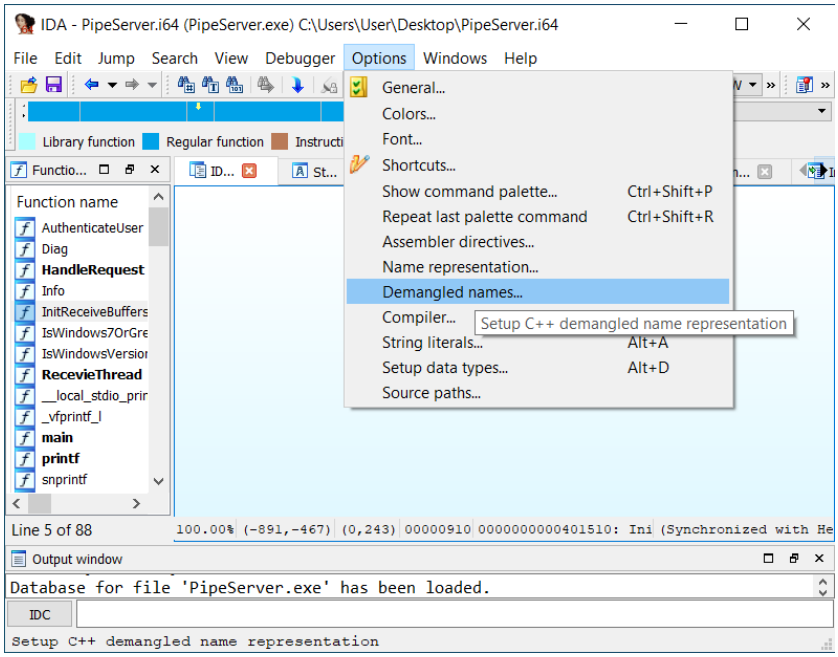
4 Outro

Vous nous ferez part de votre joie immense à résoudre ce challenge dans un rapport élégant, empreint d'une touche sobre et professionnelle. Ce dernier (ainsi que les scripts/PoC développés par vos soins) sera à renvoyer à vos bienveillants professeurs avant le dimanche 19 Janvier 2020 minuit, dernier délai.

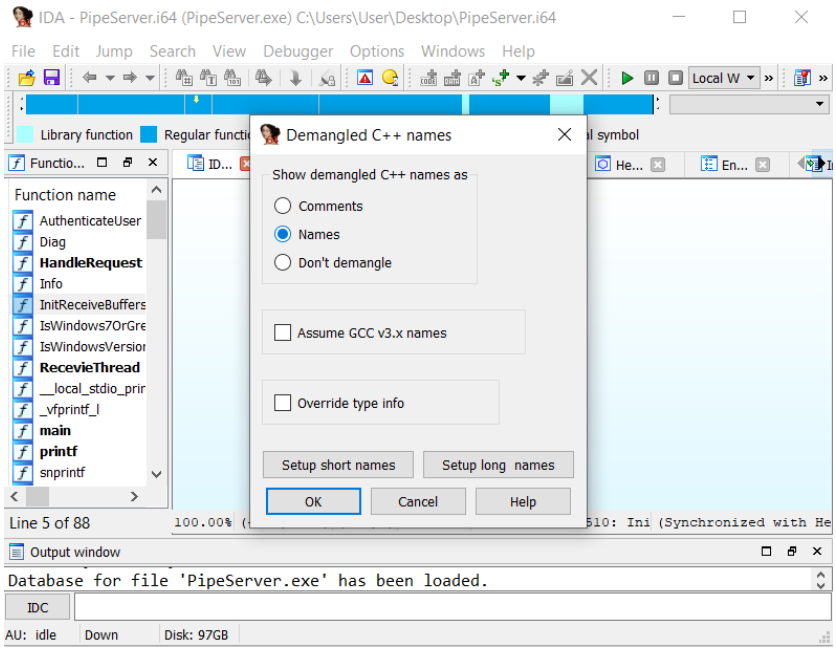
Remember:

A Tip: demangle names

Le *name mangling*[2] est une fonctionnalité des compilateurs utilisée pour encoder des informations de typage dans le nom des fonctions/variables. Une option est disponible dans `IDA` pour n'afficher que les noms simples, non *manglé*.



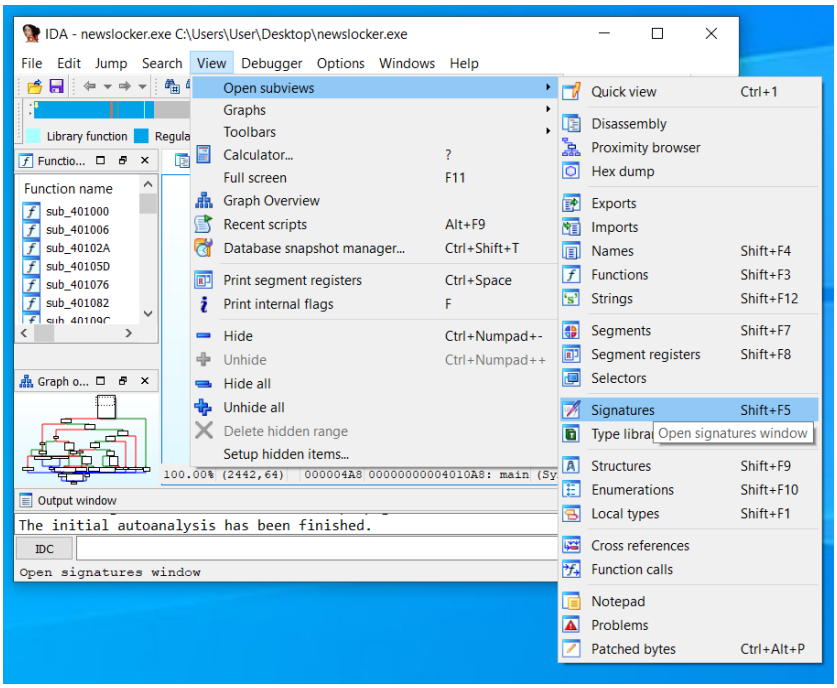
[2] https://docs.microsoft.com/fr-fr/cpp/build/reference/decorated-names?view=vs-2019



B Tip: Signature

Bien que la version *free* d'`IDA Pro` que vous utilisez dispose d'un nombre restreint de banques de signatures de fonction, vous pouvez charger le fichier *vc32rtf* afin de reconnaitre de permettre à `IDA` de reconnaitre quelques fonctions (elle apparaissent alors en bleu clair).

Enfin rappelez-vous la propriété de cohérence spatiale du code généré, si des fonctions ne sont pas reconnues mais qu'elles sont situées au milieu d'autres fonctions reconnues, il y a de très fortes chances pour que tout ce petit monde appartiennent à la même bibliothèque, et dans nos cas même runtime.


IDA - newslocker.exe C:\Users\User\Desktop\newslocker.exe
File Edit Jump Search View Debugger Options Windows Help
Open subviews
Graphs
Toolbars
Calculator... ?
Full screen F11
Graph Overview
Recent scripts Alt+F9
Database snapshot manager... Ctrl+Shift+T
Print segment registers Ctrl+Space
Print internal flags F
Hide Ctrl+Numpad+-
Unhide Ctrl+Numpad++
Hide all
Unhide all
Delete hidden range
Setup hidden items...
Quick view Ctrl+1
Disassembly
Proximity browser
Hex dump
Exports
Imports
Names Shift+F4
Functions Shift+F3
Strings Shift+F12
Segments Shift+F7
Segment registers Shift+F8
Selectors
Signatures Shift+F5
Type libra
Open signatures window
Structures Shift+F9
Enumerations Shift+F10
Local types Shift+F1
Cross references
Function calls
Notepad
Problems
Patched bytes Ctrl+Alt+P
Function name
sub_401000
sub_401006
sub_40102A
sub_40105D
sub_401076
sub_401082
Library function
Output window
The initial autoanalysis has been finished.
IDC
Open signatures window