A Szenario – Auftragsbeschreibung

Ein wichtiges Werkzeug zur Analyse und Manipulation von Binärdateien ist der *Hex-Editor*. Das ist ein Computerprogramm, mit dem sich die Bytes beliebiger Dateien als Folge von Hexadezimalzahlen darstellen und bearbeiten lassen.

In dieser Übung werden Sie in einer virtuellen Maschine mit Windows 11 einen einfachen, aber effizienten Hex-Editor installieren und mit diesem den Umgang mit Binärdateien erlernen.

Dabei werden Sie ein einfaches Programm hacken und beobachten, welche Informationen das Programm im Arbeitsspeicher ablegt.

B VM

(1) Benutzen Sie wieder Ihre virtuelle Maschine mit Windwos 11 und gehen Sie zum Snapshoi "init" zurück. Falls Sie keinen Snapshot gemacht haben, gehen Sie folgendermaßen vor:

- Löschen Sie ihre alte Windows 11 VM (*Rechtsklick = RK auf VM* / *Manage* / *Delete From Disk*)!
- Linked Clone erstellen
- Snapshot mit dem Namen *init* erstellen (*RK* / *Snapshot* / *Take Snapshot*)

(2) Ausgehende Netzwerkverbindungen erlauben
- Netzwerk zunächst auf NAT setzen: (RK auf VM / *Settings* / *Network Adapter* / *Network connection* = NAT

(3) Windows Updates aussetzen: Melden Sie sich als `junioradmin` an und setzen Sie die Windows-Updates so lange wie möglich aus! Tipp: Windows-Updates Warum sollte man in einer Produktivumgebung (und auch in Laborumgebungen, die mit dem Internet verbunden sind) *immer* aktuelle Updates einspielen?

Fügen Sie einen Screenshot in Ihr Protokoll ein!

C Installation des Hex-Editors

Leider bringt Windows keinen komfortablen Hex-Editor mit. Es gibt aber zahlreiche frei verfügbare Tools für diesen Zweck. Wir werden in dieser Laborübung **HxD** verwenden.

(4) Öffnen Sie den Web-Broswer auf Ihrem Host und gehen Sie auf: https://mh-nexus.de/de/hxd/

- Welche ist die derzeit aktuelle Version des Hex-Editors für Windows?
- Öffnen Sie in der Windows 11 VM ein Powershell-Terminal mit Administratorrechten (Wie? z.B. Tastenkombination dokumentieren!). Aktualisieren Sie die Liste der unterstützten Pakete und installieren Sie im Terminal HxD mit dem Windows-Paketverwaltung `winget`! Hinweis: Falls `winget` auf Ihrem System noch nicht vorhanden ist oder nicht funktioniert, dann laden Sie die zu installierenden Programme von deren Homepage herunter.

```
winget source update
winget install hxd
```

(5) Installieren Sie in gleicher Weise den Editor *notepad++* in der aktuellen deutschsprachigen Version herunter und installieren Sie diesen in der Windows VM. Mit welchem Kommando haben Sie die Installation durchgeführt?[1]

[1] Das Paket heißt `notepad++.notepad++`

(6) Deaktivieren Sie die Verbindung ins Internet, indem Sie wieder das Netzwerk in den VM-Settings auf Host-only setzen

D Erste Schritte mit dem Hex-Editor

(7) In notepad++: Erstellen Sie eine Datei namens `####_windows.txt` (`####` ist Ihre vierstellige Nummer) mit mehreren Textzeilen (und beliebigem Inhalt). Bei *Bearbeiten* / *Format Zeilenende*[2] wählen Sie *Windows* (falls dieses ausgegraut ist, wurde dieser Wert bereits eingestellt)! Öffnen Sie Ihre Textdatei mit HxD!

- Wozu dient die blau dargestellte oberste Zeile sowie die linke Spalte?
- Was bedeutet *Offset?*
- Erklären Sie, was in der Mitte der Ausgabe des Hex-Editors angezeigt wird!
- Was wird unter *Dekodierter Text* ausgegeben?

(8) Suchen Sie den Zeilenumbruch! Wie lautet der Hexadezimalwert des Zeichens / der Zeichen für den Zeilenumbruch?

(9) Wählen Sie nun in *notepad++* als *Format Zeilenende UNIX* und speichern Sie die Datei unter dem Namen `####_unix.txt`!
- Wie wird der Zeilenumbruch jetzt im Hex-Editor dargestellt?
- Wie speichert der Macintosh den Zeilenumbruch?

(10) Kopieren Sie die Datei `hexedittest1.exe` in Ihre Windows-VM!

Recherchieren Sie, was eine `exe`-Datei ist (kann auch zu Hause gemacht werden)?

(11) Öffnen Sie `hexedittest1.exe` mit dem Hex-Editor!
- Mit welchen beiden Zeichen (magic-number) beginnt die Datei und warum? Tipp: https://de.wikipedia.org/wiki/MZ-Datei
- Welchen Wert hat das Zeichen an der Stelle `0x945A` (hexadezimal)?

E Unser erstes gehacktes Programm

(12) Öffnen Sie eine Windows-Kommandozeile (`powershell` oder `cmd`)! Führen Sie anschließend das Programm `hexedittest1.exe` fertig aus!
- Was macht das Programm schlecht und ist daher aus Security-Sicht höchst problematisch?
- Was könnte ein böswilligerer Zuseher so alles mit Ihrem Passwort anstellen?

(13) Wir wollen nun das Programm "hacken", das heißt ohne Änderung des Quelltextes manipulieren!

- Machen Sie eine Sicherheitskopie von `hexedittest1.exe` (z.B. `hexedittest1copy.exe`)!
- Ändern Sie nun `hexedittest1.exe`, sodass die Ausgabe des Programmes lautet: *ITSI Demoprogramm – gehackt von IhrVorname IhrNachname.* Tipp: Hex-Editor! Achtung! **Die Dateigröße darf sich dabei nicht ändern** – Sie müssen daher den Text überschreiben (und nicht einfügen)![3]
- Starten Sie das manipulierte Programm und dokumentieren Sie Ihren Hack mit einem Screenshot im Protokoll!

[2] in englischen Versionen von notepad++ heißt dieser Menüpunkt *Edit / EOL convention*
[3] Den Grund dafür lernen Sie in Systemtechnik / Grundlagen der Informatik

F Manipulation am laufenden Programm

Abschließend wollen wir untersuchen, welche Informationen das ausgeführte Programm im Arbeitsspeicher ablegt. Wie Sie (hoffentlich) aus dem Systemtechnik-Unterricht bereits wissen, werden die Daten eines ablaufenden Programmes (z.B. Werte von Variablen) im Arbeitsspeicher abgelegt[4]. Sie haben (als angehender Security-Experte wahrscheinlich mit Entsetzen) festgestellt, dass unser Testprogramm die eingegebenen Passwörter wenig vertraulich behandelt.

(14) Wir werden nun mit dem Hex-Editor einen Speicher-Dump[5] anfertigen und auf diese Weise das eingegebene Passwort ermitteln!

- Starten Sie `hexedittest1.exe` mittels Doppelklick im Explorer und geben Sie als Passwort Ihren Familiennamen ein, sodass das Programm *"Bitte Taste druecken, um das Programm zu beenden!"* anzeigt! Beenden Sie das Programm jetzt *noch nicht*!
- Öffnen Sie den Hex-Editor, drücken Sie auf das Icon für *Arbeitsspeicher öffnen* und wählen Sie `hexedittest1.exe` aus!
- Suchen Sie nach dem eingegebenen Passwort im Hex-Editor (*Strg-F*). Fügen Sie einen Screenshot in Ihr Protokoll ein!
- Ändern Sie nun das Passwort im Hex-Editor und schreiben Sie die Änderung in den Arbeitsspeicher zurück (Tipp: *Strg-S*)
- Bonus: Wie viele Zeichen sollten Sie maximal im Hex-Editor für das geänderte Passwort eingeben? Warum? Tipp: sehen Sie sich den Sourcecode(`hexedittest1.cpp`) an!
- Beenden Sie nun das Testprogramm durch einen Tastendruck und beschreiben Sie, was passiert ist!

(15) Sie können auch mit Windows-Bordmitteln (als Administrator) in den Speicher anderer Programme blicken:

- Starten Sie `hexedittest1.exe`, geben Sie als Passwort Ihren Familiennamen ein und beenden Sie das Programm *noch nicht*!
- Starten Sie den *Taskmanager*[6] (Welchen Shortcut verwenden Sie dafür?)
- Erstellen Sie einen Memory-Dump = *Speicherabbilddatei*! (Tipp: Rechtsklick auf Prozess)
- Wo (absoluter Pfad) wird der Speicherauszug abgelegt?
- Suchen Sie in der erzeugten Dump-Datei mittels Hex-Editor das eingegebene Passwort! Fügen Sie einen Screenshot in Ihr Protokoll ein!

(16) **Achtung:** Memory-Dumps werden häufig automatisch nach einem Programmabsturz generiert, um EntwicklerInnen einen Anhaltspunkt zur Analyse des Problems zu geben. Leider bieten solche *Crash-Dumps* oft auch eine gute Gelegenheit für Hacker. Einem derartigen Angriff mit besonders schwerwiegender Auswirkung ist sogar die Firma Microsoft zum Opfer gefallen: siehe Datei im Anhang oder https://www.heise.de/select/ct/2023/22/2325014433560180576.

Bonus: Erklären Sie mit eigenen Worten, was bei diesem geschilderten Angriff vorgefallen ist.

G Analyse des Source-Codes

(17) Nachdem wir unser erstes Programm gehackt haben, analysieren wir nun den Quellcode von `hexedittest1.exe`.Dieses kleine Programm wurde in *C* geschrieben. Im SEW-Unterricht haben Sie zwar die

[4] Ausnahmen dafür sind jene Variablen, die von einem optimierenden Compiler ausschließlich in ein Register gepfercht wurden
[5] das ist eine Kopie oder ein Auszug eines Speicherinhaltes
[6] Vollprofis verwenden stattdessen den weitaus mächtigeren *Process Explorer* von Sysinternals

Programmiersprache *Java* gelernt – da diese sehr stark von C beeinflusst wurde, wird Ihnen die Analyse des Source-Codes nicht schwerfallen.

(18) Recherchieren Sie (können Sie auch in Ruhe zu Hause erledigen):

- Was ist ein Preprocessor?
- Was bewirkt die Preprocessor-Anweisung `#include`?
- Was bewirkt die Preprocessor-Anweisung `#define`?

(19) Fügen Sie die Quelldatei `hexeditest1.c` hier im Protokoll ein (eine `nichtproportionale = monospace-"Schreibmaschinenschrift"` verwenden!) und kommentieren Sie die wesentlichen Programmzeilen. Auch diese Aufgabe können Sie zu Hause machen.

Viel Spaß!