ATIS - Software Reverse Engineering

Prof. Dr. Christian Dietrich
dietrich@internet-sicherheit.de

5 Lab: Dynamic Code Analysis

Hier werden (ab der nächsten Seite) ergänzende Lösungshinweise zur Verfügung gestellt, die Ihnen helfen sollen, die Aufgabe zu bearbeiten. Spoiler alert: Allerdings werden damit auch Teillösungen, die Sie sich ansonsten vielleicht selbst erarbeitet hätten vorweggenommen. Um einen maximalen Lernerfolgt zu erzielen, sollten Sie nur schrittweise diese Lösungshinweise konsultieren, etwa wenn Sie aus Ihrer Sicht zu lange an einem Schritt hängen.

Ergänzende Lösungshinweise

- Nutzen Sie am besten die SRE-VM zur Analyse. In der VM sind bereits ein paar Tools und Abhängigkeiten erfüllt, die die Analyse ansonsten unnötig erschweren würden (es ist aber durchaus möglich, die Datei auch in anderen Umgebungen zu analysieren).
- Befassen Sie sich zunächst damit, herauszufinden welche Art an Datei Sie vor sich haben. Es handelt sich um eine ELF-Datei.
- Mit dem Tool `readelf` können Sie die Eigenschaften einer ELF-Datei anzeigen. Untersuchen Sie, für welche Architektur die ELF-Datei kompiliert wurde und ob es für 32- oder 64-bit kompiliert wurde.
- Erinnern Sie sich an das Diagramm, das uns seit Beginn der Vorlesung begleitet und den Prozess des Reverse Engineerings beschreibt. Versuchen Sie herauszufinden, ob das vorliegende Binary statisch oder dynamisch kompiliert wurde. Gibt es Bibliotheken, die verwendet werden?
- Versuchen Sie als nächstes eine statische Code-Analyse vorzunehmen. Sie können sich dazu beispielsweise einige der Tools, die in der Vorlesung genannt wurden vornehmen, etwa IDA free (`https://www.hex-rays.com/products/ida/support/download_freeware.shtml`) oder die Demo-Version von Binary Ninja (`https://binary.ninja/demo/`).
- Betrachten Sie kritisch, ob die statische Code-Analyse hier sinnvoll erscheint.
- Möglicherweise ist die Datei obfuskiert. Erinnern Sie sich an die Inhalte der Vorlesung und überlegen Sie, welche Analysetechnik im gegebenen Fall vielversprechend aussieht.
- Ganz genau, die dynamische Code-Analyse ist hier vermutlich aussichtsreicher. Bedenken Sie, dass Sie es potentiell mit zerstörerischem Code zu tun haben könnten. Es ist daher jetzt ein guter Zeitpunkt, um einen Snapshot Ihrer VM zu machen, zu dem Sie - falls die Malware ihre Zerstörungsfunktion aktiviert - zurückkehren können.
- Versuchen Sie nun, das Programm zu starten. Vermutlich erhalten Sie eine nicht unbedingt aussagekräftige Fehlermeldung. Warum? Haben Sie alle Abhängigkeiten des Programms erfüllt (Stichwort Bibliothekscode)?
- Das Programm benötigt die Bibliothek openssl. Vermutlich haben Sie die 64-bit Version von openssl bereits installiert, aber haben Sie auch die 32-bit Version, die das vorliegende Programm benötigt? Installieren Sie die 32-bit Variante der openssl-Bibliothek:
  `sudo apt install libssl-dev:i386`
  Sie sehen, eine Schwierigkeit der dynamischen Code-Analyse besteht darin, die Zielumgebung insoweit nachzubauen, dass alle Abhängigkeiten erfüllt sind. Das kann sich natürlich auch auf andere Aspekte beziehen, etwa bestimmte Verzeichnisse, die das Programm erwartet etc. Aber keine Sorge, in dieser Aufgabe gibt es keine weiteren solcher Tricks.
- Wenn Sie das Programm auf der Kommandozeile erfolgreich ausführen können, sind Sie schon mal ein Stück weiter. Überlegen Sie nun und probieren Sie nun Techniken zur dynamischen Code-Analyse aus, zum Beispiel Debugging. Was passiert, wenn Sie das Programm in einem Debugger ausführen?
- Scheinbar erkennt das Programm, wenn es in einem Debugger ausgeführt wird und wehrt sich dagegen. Was könnte man hier tun?
- Stichwort: Patching. Versuchen Sie die Stelle zu finden, die dafür verantwortlich ist, dass sich das Programm gegen einen Debugger wehrt. Überlegen Sie nun, ob Sie hier den Maschinencode modifizieren können, sodass der Check nicht mehr ausgeführt wird oder zu Ihren Gunsten entscheidet.
- Wenn das Debugging Ihnen an der Stelle nicht weiterhilft, können Sie auch andere Techniken in Betracht ziehen, zum Beispiel Dynamic Binary Instrumentation mit Hilfe von Intel Pin.