Vortessence Rule Engine

Studiengang: BSc in Informatik | Vertiefung: IT-Security
Betreuer: Prof. Dr. Endre Bangerter
Experte: Dr. Igor Metz (Glue Software Engineering AG)

Die Memory Forensik, also die Analyse von Abbildern des RAM Speichers, ist eine Schlüsseltechnik für die Detektion und Analyse von Cyberattacken. Die Idee ist, dass Angreifer unvermeidliche Spuren im RAM Speicher hinterlassen. Dies sind beispielsweise unbekannte Prozesse, oder subtile Systemmodifikationen. Die Memory Forensik erlaubt es, diese Spuren, bei denen es sich um Anomalien handelt, zu finden und somit Angriffe zu detektieren und zumindest teilweise zu verstehen.

Ausgangslage

Vortessence ist ein Open-Source-Projekt des Security Engineering Labs der BFH und hat das Ziel, Memory Images automatisiert zu analysieren. Die Analyse der Memory Images geschieht mittels Whitelists, die mit Hilfe nicht kompromittierter Systeme erstellt werden. Die dafür benötigten Artefakte werden durch das Open Source Memory Forensik Framework Volatility aus den verschiedenen Memory Images extrahiert. Memory Images potentiell kompromittierter Systeme werden in Vortessence gegen die erstellten Whitelist-Einträge verglichen, um damit Anomalien zu erkennen.

Die oben beschriebene Funktionsweise kann in gewissen Fällen zu False-Positives führen. Beispielsweise führt ein nicht gewhitelisteter Registry-Key eines neu installierten Programms zu einer Detektion. Also ist die Anzahl der False-Positives unter anderem von der Grösse der Whitelist abhängig. Weiter gibt es auch legitime Gründe für Software sogenannte Hooks auszuführen oder eine Memoryregion als ausführbar zu markieren, was schlussendlich durch die Volatility Plugins Apihooks, Malfind etc. zu vielen False-Positives führt.

Suchen nach Malware im RAM

Ziele

Das Ziel der Arbeit war die Verbesserung der Detektionsqualität von Vortessence, indem weiterführende Ansätze verfolgt werden, um die Echtheit von Code Injections und Modifikationen im Code zu erkennen. Dazu soll unter anderem ein Algorithmus entwickelt werden, der Code inhaltlich vergleicht. Damit können identische PE-Files (exe, dll) auf Gleichheit überprüft und somit durch Malware vorgenommene Modifikationen erkannt werden. Mit diesem Codevergleich sollen identische Injections in verschiedenen Prozessen gefunden werden (Cross-Process Korrelation). Durch eine detaillierte Untersuchung mittels Vorher-Nachher-Memory-Images soll die Effektivität der neu implementierten Funktionen gemessen werden. Alle neuen Funktionen sollen im Rahmen dieser Bachelor-Thesis in der bestehenden Konsolenanwendung aufgerufen, sowie im Web-Frontend von Vortessence sinnvoll dargestellt und visualisiert werden.

Ergebnisse

Durch einen neu implemenentierten Codevergleich können beispielsweise durch eine Malware injizierte Apihooks erkannt werden. Mit einem Korrelationsalgorithmus war es möglich, die Anzahl der False-Positives von Apihooks / Malfind zu einem beträchtlichen Teil zu reduzieren. Ein Algorithmus zur Cross-Process Korrelation, basierend auf dem oben genannten Codevergleich, identifiziert im selben Memorydump alle identischen Code-Fragmente, die in verschiedenen Memoryregionen injected wurden. Die neuen Funktionen lassen sich alle über die von uns implementierten Vortessence Rules aufrufen. Deren Resultate werden in der bestehenden Weboberfläche visualisiert und können von einem Memory Forensiker interaktiv analysiert werden.

Marco Berger

Andy Li Fei Pollari