DuD
Datenschutz und Datensicherheit

Auffinden von verschleierter Malware

Einsatz der heuristischen Analyse

Marcel Lehner, Eckehard Hermann

Malware-Autoren setzen zunehmend Verschleierungsmechanismen wie Komprimierung, Verschlüsselung oder sich selbst modifizierenden Programmcode ein, welche das Auffinden der Malware durch herkömmliche Virenerkennungsverfahren wie zum Beispiel die signaturbasierte Suche sehr mühsam und teilweise sogar erfolglos erscheinen lassen. Ein erfolgversprechender Ansatz, um auch unbekannte Viren, Würmer oder Trojaner zu erkennen, ist die heuristische Analyse

Marcel Lehner

Absolvent des FH-Magisterstudiengangs „Sichere Informationssysteme“, der FH-OÖ Studienbetriebs GmbH Campus Hagenberg.
E-Mail: marcel.lehner@sectec.at

Eckehard Hermann

Hochschullehrer an der FH-OÖ Studienbetriebs GmbH Campus Hagenberg, FH-Studiengänge „Computer- und Mediensicherheit“ und „Sichere Informations-systeme“
E-Mail: eckehard.hermann@fh-hagenberg.at

Einleitung

Datenverlust, Produktionsausfälle und verärgerte Anwender sind nur einige Auswirkungen, die Viren, Würmer und Trojaner verursachen. Auch kriminelle Organisationen zeigen vermehrt ein Interesse daran, wirtschaftlichen Nutzen aus dem Einsatz dieser, mit dem Überbegriff Malware bezeichneten gefährlichen Software zu erzielen.

Malware-Autoren setzen zunehmend Verschleierungsmechanismen wie Komprimierung, Verschlüsselung oder sich selbst modifizierenden Programmcode, wie z.B. metamorphe Malware ein, um ihre schädlichen Dateien vor einer Erkennung durch Virenscanner zu schützen.

Die dadurch entstehende Komplexität der Verschleierungsarten von Malware verlangt von Antivirenherstellern neben der bisherigen statischen Analyse, welche sich primär mit der Erkennung von Byte-Folgen beschäftigt, die Unterstützung von semantischen Analyseverfahren. Besonders die verhaltensanalytischen Eigenschaften der Heuristik machen diese zu einem geeigneten Kandidaten zum Erkennen von verschleierter, gefährlicher Software.

1 Malware

Bei dem Begriff Malware handelt es sich um ein Kunstwort, welches sich aus den ersten drei Buchstaben von **mal**icious (boshaft) und den letzten vier Buchstaben von Soft**ware** zusammensetzt. Malware ist Software, welche mit dem Ziel entwickelt wurde, unerwünschte und in der Regel schädliche Funktionen auszuführen. Prinzipiell lassen sich drei unterschiedliche Hauptarten von Malware unterscheiden:

- Ein *Virus* ist eine sich selbst vervielfältigende Software, welche sich in andere Programme oder Dokumente einschleust und bei jeder Ausführung der infizierten Anwendung gestartet wird.
- *Würmer* sind den Viren sehr ähnlich. Der Unterschied liegt darin, dass sie sich über das Intra- oder Internet replizieren und meist keine Interaktion des Benutzers benötigen.
- Unter den möglichen Schadensfunktionen in Viren und Würmern ist das so genannte *Trojanische Pferd* besonders gefährlich. Hierbei handelt es sich um Programme, die dem Benutzer eine nützliche Funktion vortäuschen. Allerdings enthalten diese Programme schädliche Routinen, die ohne die Zustimmung oder das Wissen des Benutzers mit ausgeführt werden.

Viele Viren enthalten eine Mischung aus mehr als einer Schadensfunktion, was eine eindeutige Klassifizierung schwierig macht.

Neben diesen Vertretern fallen u. a. auch Exploits, Spyware, Backdoors, Rootkits unter den Begriff Malware. Diese können ebenfalls, wie ihre Artgenossen, verschleiert werden.

2 Erkennungsmechanismen

Da ein Anwender in der Regel nicht freiwillig einen für sein System schädlichen Programmcode ausführt, werden schadhafte Programme von ihren Autoren verschleiert.

Theoretisch kann jede verschleierte Software erkannt werden, wenn nur genügend Zeit und Ressourcen zur Verfügung stehen, was jedoch oft nicht der Fall ist. Immer besser verschleierte und sich schneller verbreitende Malware hat dazu beigetragen, dass Antivirenhersteller verschiedene Verfahren entwickelt haben, um verschleierte Malware zu erkennen und Anwender und deren Rechner zu schützen.

2.1 Virus-Signaturen

Die am weitesten verbreitete Art der Malwareerkennung ist das so genannte signaturbasierte Verfahren. Bei diesem Verfahren wird aus dem Binärcode der Malware eine eindeutige Byte-Folge extrahiert. Diese kann als Fingerabdruck oder auch Signatur der Malware angesehen werden. Die erstellte Signatur wird in die Signaturdatenbank der Erkennungssoftware, z.B. eines Virenscanners, eingespielt. Diese wiederum untersucht sämtliche zu überprüfenden Dateien nach dieser Signatur. Wird die Signatur in einer Datei gefunden, kann mit hoher Wahrscheinlichkeit von einer Infektion ausgegangen werden.

Ein großer Nachteil der signaturbasierten Malwareerkennung liegt darin, dass Computer in der Zeit zwischen dem Auftreten der neuen Malware und der Erstellung bzw. dem Einspielen einer passenden Signatur in die Signaturdatenbank der Erkennungssoftware ohne weitere Sicherheitsvorkehrungen ungeschützt sind.

2.2. Neuronale Netze

Neuronale Netze setzen auf künstliche Intelligenz und stellen lernfähige Verfahren dar, die Zusammenhänge mit Hilfe vorhandener Daten schrittweise erlernen können. Hierbei macht man sich die Eigenschaft von Neuronalen Netzen zu nutze, die anhand von Beispielen ein bestimmtes Muster erlernen können und anschließend mit einer gewissen Wahrscheinlichkeit dieses Muster wieder erkennen.

Je mehr Material (bekannte Malware mit einem bestimmten Verhaltensmuster) in der Lernphase eingespielt wird, desto höher ist die Erkennungsrate. Da Neuronale Netze nur eine Aussage mit einer gewissen Wahrscheinlichkeit ermöglichen, kann keine 100 %ige Erkennung garantiert werden. In der Praxis zeigen sich Neuronale Netze jedoch effektiv beim Erkennen von Malware, die der im Training verwendeten gefährlichen Software vom Verhaltensmuster, wie beispielsweise dem Aufruf bestimmter Systemfunktionen, entspricht.

Hierbei kann es sich um Viren handeln, die lediglich eine neuere Version eines schon bekannten Virus sind oder ein schon von einem anderen Virus bekanntes Verschleierungsverfahren nutzen.

2.3 Integritätsprüfung

Um sich selbst zu vervielfältigen muss sich ein Virus in andere Programme oder Dokumente „hineinkopieren“ und dabei Änderungen an der Wirtsdatei vornehmen. Programme, die diese Änderungen erkennen, können eine mögliche Infizierung feststellen.

Ein Programm dieser Art bildet eine einmalige Checksumme über jede Datei eines nicht infizierten Systems und speichert diese in einer Datenbank. In periodischen Abständen werden die Checksummen neu gebildet und mit den bereits in der Datenbank gespeicherten Werten verglichen. Sind diese ungleich, liegt eine Änderung der jeweiligen Datei vor.

Mit Hilfe dieser Technik ist es möglich, bekannte und unbekannte Malware, auch ohne Erstellen und Pflegen einer Signaturdatenbank, zu erkennen. Das Problem dieses Verfahrens liegt jedoch darin, dass Dateien und Programme nicht nur durch gefährliche Software geändert werden, sondern auch von legalen Anwendungen oder direkt durch den Anwender, zum Beispiel bei der Einspielung neuer Versionen („Updates“). Auch wird Malware nur beim Prüfen der Checksummen erkannt, zu einem Zeitpunkt, zu dem die Schadfunktion möglicherweise bereits ausgeführt wurde.

Daher liegt es zum einen in der Hand des Integritätsprüfers und zum anderen in der Hand des Anwenders zu entscheiden, welche Änderungen legal und welche möglicherweise durch Malware entstanden sind.

2.4 Behavior Blocker

Im Vergleich zur Integritätsprüfung ist das „Behavior-Blocking“ ein komplett anderer Ansatz zur Malwareerkennung. Bei diesem Erkennungsmechanismus werden Dateiaktivitäten und deren Verhalten überwacht. Es wird zunächst festgelegt, welche Aktionen und Änderungen von und an Dateien und Programmen erlaubt oder verboten sind.

Obwohl Behavior Blocker sämtliche Vorgänge auf einem System in Echtzeit überwachen, schützen sie nur bedingt gegen bekannte und unbekannte Malware.

Behavior Blocker können bei einer Verhaltensanomalie nicht feststellen, ob es sich tatsächlich um Malware, und wenn, um welche Art es sich handelt. Es wird zwar erkannt, dass eine unerlaubte Aktion ausgeführt wird, dennoch kann nicht bestimmt werden, ob es sich um einen Virus, Wurm oder eine andere Art von gefährlicher Software oder aber um eine ungewöhnliche, aber zulässige Aktion handelt.

3 Heuristische Analyse

Die heuristische Analyse von Malware wird bereits seit Beginn der neunziger Jahre eingesetzt und ist der signaturbasierten Suche zum Teil ähnlich. Der Unterschied liegt darin, dass bei der heuristischen Analyse nicht nach einzigartigen Signaturen gesucht wird. Statt dessen werden bestimmte Funktions- und Prozessaufrufe eines Betriebssystems auf spezielle Verhaltensmuster (Anomalien) hin untersucht, welche in herkömmlichen Anwendungen normalerweise nicht zu finden sind. Daraus ergibt sich die Möglichkeit, neue, auch noch unbekannte Malware zu erkennen.

Antivirenhersteller haben zwei verschiedene Ansätze der heuristischen Analyse zur Erkennung von Malware entwickelt: die statische und die dynamische Heuristik.

- Bei der *statischen Heuristik* wird wie bei der signaturbasierten Virenerkennung nach zuvor festgelegten Eigenschaften gesucht. Sind es bei Virensignaturen eindeutig festgelegte Byte-Folgen, mit denen verglichen wird, so sind es bei der statisch-heuristischen Analyse festgelegte Strukturen im Programmcode, die untersucht werden.
- Im Unterschied dazu wird bei der *dynamisch-heuristischen Analyse* das Verhalten eines Programms untersucht. Dazu muss es allerdings ausgeführt werden, was zum Schutz des Systems in einer gesicherten, oft emulierten Umgebung stattfindet.

3.1 Statische heuristische Scanner

Statische heuristische Scanner setzen verschiedene Methoden ein, um das Verhalten von Programmen zu analysieren. Eines dieser Verfahren ist der signaturbasierten Malwareerkennung sehr ähnlich. Tatsächlich sind es oft dieselben Signaturen, nur mit einer anderen Bedeutung. Auch hier wird nach einer bestimmten Byte-Folge gesucht. Allerdings setzt sich diese Byte-Folge aus einer bestimmten Abfolge von Instruktionen zusammen. Somit beschreibt die Signatur ein bestimmtes Verhalten.

Neben den verhaltensanalytischen Signaturen gibt es noch andere Hilfsmittel für statisch-heuristische Scanner. Zum Beispiel verwendet verschlüsselte Malware beim Ausführen einer infizierten Anwendung Systemfunktionen, um eigene Programmteile zu entschlüsseln. Kennt ein Scanner diese Funktionen, so ist es ihm möglich, auch die verschleierte Malware zu entdecken.

3.2 Dynamische heuristische Scanner

Statisch-heuristische Scanner sind, wie ihre Verwandten, die signaturbasierten Scanner, auf die Erkennung von bereits bekannter Malware ausgerichtet. Erfolgreich hat sich dieses Verfahren jedoch, ähnlich wie die Neuronalen Netze, auch bei der Auffindung unbekannter, verschleierter Malware erwiesen, die von einem bereits bekannten Virenstamm abgeleitet wurde.

Teilweise ist es mit der statisch-heuristischen Analyse auch möglich, einfache Verschlüsselungsroutinen zu entdecken. Doch viele der statisch-heuristischen Scanner versagen z.B. beim Einsatz von den weiter unten betrachteten polymorphen oder metamorphen Verschleierungsmethoden.

Daher nutzen dynamisch-heuristische Scanner Code-Emulierung, um Informationen über nicht vertrauenswürdige Anwendungen zu erhalten. Hierbei lädt der Scanner die zu untersuchende Anwendung in eine virtuelle Umgebung und simuliert die Ausführung dieser Datei. Während der Simulation kann so das Verhalten der zu testenden Datei vom dynamischen Scanner erfasst und katalogisiert werden. Dabei kann der Scanner z.B. alle Interrupt-Aufrufe, die das verdächtige Programm an das Betriebssystem schickt, überwachen.

Besonders bei stark verschlüsselter und polymorpher Malware hat das dynamische Verfahren einen großen Vorteil gegenüber dem statischen. Hier können sämtliche Ver- und Entschlüsselungsschritte beobachtet werden. Weiter kann anschließend die entschlüsselte Software, der eigentliche Schadcode, untersucht werden.

Der Nachteil des dynamischen Verfahrens liegt unter anderem in der Geschwindigkeit der Scanner. Eine Simulation kann sehr zeitintensiv sein. Hingegen ist die Suche nach Signaturen in bestimmten Bereichen einer Anwendung äußerst schnell.

Ein dynamischer Scanner erlaubt dem Analyseobjekt, sich frei im emulierten System zu bewegen. Daher spielt es keine Rolle, wie das Programm und dessen Struktur und Logik zum Erreichen des Zieles aufgebaut ist. Spätestens bei der Kommunikation mit dem virtuellen Betriebssystem ist klar, welche Absichten das Objekt verfolgt.

4 Verschleierungsmechanismen

Gute Programmverschleierung ist nicht nur schwierig zu erkennen, sondern auch schwierig zu erreichen. Im Folgenden werden einige der am Häufigsten anzutreffenden Verfahren vorgestellt und deren Erkennung durch heuristische Analyseverfahren betrachtet.

4.1 Code Reordering

Dieses Verfahren ist eine sehr einfache Art der Verschleierung. Bei dieser Methode werden die Instruktionen im Quelltext in einer neuen Reihenfolge angeordnet. Um die Semantik nicht zu ändern und den ursprünglichen Ablauf des Programms zu gewährleisten, werden Sprungbefehle eingesetzt.

Ein heuristischer Ansatz zum Erkennen von Code Reordering ist der Einsatz von Control Flow Graphs (CFG) [1]. Zunächst werden sämtliche Befehle, die das virale Verhalten der ursprünglichen Malware definieren, in der richtigen Reihenfolge festgehalten. Daraus wird dann ein CFG erstellt. Dieser beschreibt den genauen Ablauf, den eine Anwendung erfüllen muss, um als gefährliche Software eingestuft zu werden.

Wird eine nicht vertrauenswürdige Anwendung durch das Emulationsmodul eines heuristischen Scanners ausgeführt, kann dieser sämtliche Instruktionen sequenziell abarbeiten und mit dem CFG vergleichen. Dabei ist es unerheblich, wie viele Sprunganweisungen durch das Code Reordering eingefügt wurden. Für unbekannte Viren-, Würmer- und Trojaner-Stämme müssen nach ihrem ersten Auftreten neue CFGs definiert und durch ein Update in die Erkennungssoftware eingespielt werden.

4.2 Entry Point Obfuscation

Eine besonders schwer erkennbare Methode der Verschleierung ist die Entry Point Obfuscation (EPO) Technik. Herkömmliche Malware ändert die Einsprungadresse eines Programms nach der Infizierung meist so, dass sie auf den viralen Code zeigt. Bei verschlüsselter oder komprimierter Malware zeigt die Einsprungadresse auf den Anfang der Entkomprimierungs- bzw. Entschlüsselungsroutine. Genau diese Technik macht sich die Erkennungssoftware zunutze. Sie analysiert nur den Bereich um die Einsprungadresse, bzw. das Ziel dieser Adresse auf maliziösen Code hin. Der Grund dafür liegt darin, dass eine Untersuchung der gesamten Anwendung zu zeitaufwändig wäre.

EPO-Malware ändert nicht die Einsprungadresse der zu infizierenden Anwendung. Statt dessen wird ein zufälliger Bereich im Programmcode so umgeschrieben, dass er auf den viralen Teil verweist. Bei der Analyse von EPO-Malware kann somit nicht mehr auf die Untersuchung der Einsprungadresse als Hilfsmittel zurückgegriffen werden, da diese beim vorliegenden Verschleierungsverfahren nicht geändert wird. Aus diesem Grund müsste jede Datei vollständig untersucht werden, um den Virencode zu finden.

Merkmal von EPO-Verschleierungen ist, dass die Infektionen hinter der eigentlichen Einsprungadresse durchgeführt werden, und der Virencode in der letzten Sektion einer infizierten Datei eingebettet wird. Da ein Sprungbefehl z.B. durch die Instruktion *call* charakterisiert wird, muss der heuristische Scanner zunächst nach dem zugehörigen Maschinenbefehl für die *call* Instruktion suchen. Hat er diesen gefunden, kann er anhand der nachfolgenden vier Bytes den Bereich der Sprungadresse berechnen. Liegt dieser in der letzten Sektion, könnte es sich um einen Virenbefall handeln und weitere Schritte können eingeleitet werden [2].

Ein Problem liegt jedoch darin, dass der Sprungbefehl beliebig oft im Code der Malware eingefügt worden sein kann. Das wiederum führt zu einer rechenintensiveren Untersuchung der Malware, da für jedes gefundene Byte die Zieladresse berechnet werden muss.

EPO kann als eine der stärksten Verschleierungen angesehen werden. Selbst der Einsatz von dynamisch-heuristischen Scannern kann an ihr erfolglos bleiben. Auch eine Emulierung von nicht vertrauenswürdigen Anwendungen kann nicht gewährleisten, dass die Einsprungadresse zur eigentlichen Malware innerhalb der Emulationszeit oder überhaupt aufgerufen wird.

4.3 Packing

Da Würmer wie der W32/Blaster [6] in einer Hochsprache (HLL) programmiert wurden und dadurch einen umfangreichen Programmcode besitzen, setzen Virenautoren Komprimierungsprogramme ein. Diese bringen zwei Vorteile für den Angreifer mit:

- Zum einen reduzieren sie die Größe der gefährlichen Software,
- zum anderen wird durch das Komprimieren die Binärdarstellung einer Datei geändert und dadurch die Malware, je nach verwendeten Komprimierungsverfahren, verschieden stark verschleiert.

Mittlerweile werden 90 % aller 32Bit-Würmer mit einem Packer wie UPX oder ASPACK komprimiert [3].

Zur Zeit stehen den Virenautoren schon einige hundert verschiedene Versionen von Komprimierungsprogrammen zur Verfügung. Daneben können auch eigene Versionen vom Virenautor erstellt werden, die es der Erkennungssoftware erschwert, komprimierte Malware zu erkennen und zu analysieren.

Durch den Einsatz eines Emulators, der die nicht vertrauenswürdige Datei zur Laufzeit in einer gesicherten Umgebung ausführt und deren Dekomprimierung beobachtet, ist es möglich, gepackte Malware zu analysieren. Die nicht vertrauenswürdige Anwendung wird in einem Emulator ausgeführt und im Speicher mit dem mitgeführten Dekomprimierungsalgorithmus entpackt. Sind genügend Informationen bzw. Bytes im Speicher entpackt, kann der Scanner diese mit dem herkömmlichen Viren-Signatur-Verfahren analysieren. Der Einsatz eines Emulators hat an dieser Stelle den Vorteil, dass der Antivirensoftwarehersteller den von der Malware zum Dekomprimieren verwendeten Algorithmus nicht kennen muss.

Allerdings kann dieses Verfahren durch das Emulieren jedes einzelnen Arbeitsschrittes sehr rechenintensiv werden.

4.4 Verschlüsselung

Ein häufig eingesetzter Verschleierungsmechanismus ist die Verschlüsselung. Die Verschlüsselung wird neben dem Verschleiern der Code Signatur auch verwendet, um eine Disassemblierung und die damit verbundene Analyse der gefährlichen Software zu verhindern.

Verschlüsselte Malware besteht aus einer Entschlüsselungsroutine und einem verschlüsselten Teil, der die bösartige Software enthält [4]. Sobald ein Benutzer ein Programm dieser Art ausführt, wird zunächst die Entschlüsselungsroutine ausgeführt, die den Virus oder Wurm entschlüsselt. Danach übernimmt dieser die Kontrolle und infiziert weitere Anwendungen und Dateien. Dabei wird bei jeder Infektion eine Kopie des entschlüsselten Teils und der Entschlüsselungsroutine erstellt. Anschließend wird der entschlüsselte Teil erneut verschlüsselt und samt Entschlüsselungsfunktion an die zu infizierende Datei angehängt.

Ein heuristischer Ansatz zum Entdecken von verschlüsselnder Malware ist zum Beispiel die Analyse der nicht vertrauenswürdigen Anwendung auf das Vorhandensein des Karatsuba-Multiplikations-Algorithmus oder ähnlicher rekursiver Verfahren. Bei dem Karatsuba-Verfahren handelt es sich um einen effektiven Algorithmus zur Multiplikation von zwei großen Zahlen, wie sie in der Public-Key Kryptographie oft benötigt werden und somit auch in den Ver- und Entschlüsselungsroutinen kryptographischer Malware zum Einsatz kommen. Angreifer setzen dieses Verfahren ein um Daten auf dem infizierten System zu verschlüsseln und den privaten Schlüssel, welcher für die entschlüsselung benötigt wird, anschließend an das Opfer zu verkaufen.

Das rekursive Verhalten eines Verschlüsselungsalgorithmus kann wiederum mit Hilfe der heuristischen Analyse erkannt werden, indem der Control Flow Graph (CFG) der Malware aus deren Binärcode erstellt und untersucht wird. [4]

4.5. Polymorphe Malware

Polymorphie heißt Vielgestaltigkeit. Unter polymorpher Malware versteht man daher Malware, die in verschiedenen Versionen mit derselben Wirkung auftritt. Dabei verwendet polymorphe Malware unterschiedliche Verschlüsselungsmethoden und arbeitet mit mehreren unterschiedlichen Schlüsseln. Da sich mit jeder neuen Infektion die Entschlüsselungsfunktion ändern kann, ist es bei diesem Verschleierungsverfahren schwierig, eine passende Virensignatur zu erstellen. Eine Untersuchung des polymorphen Tremor Virus [7] zeigt, dass dieser in der Lage ist, sechs Milliarden verschiedene Formen anzunehmen.

Polymorphe Malware setzt neben der Verschlüsselung noch andere Verschleierungsverfahren ein, wie die so genannte Garbage Insertion, bei der sinnlose Instruktionen eingefügt werden, um den Grad der Verschleierung zu erhöhen.

Ein weiterer Ansatz der Polymorphie ist das Einfügen von sinnlosen Sprungbefehlen und Permutation. Hierbei wird die Malware in verschiedene Funktionen aufgeteilt und dann in einer zufälligen Reihenfolge angeordnet.

Die Suche nach polymorpher Malware mit herkömmlichen Virensignaturen hat sich bald als umfangreiches Unterfangen erwiesen. Der Grund dafür liegt darin, dass eine neue Signatur für jede geänderte Version eines polymorphen Virus entwickelt werden muss.

Effizienter ist der Einsatz der heuristischen Analyse. Wenn z.B. ein Programm verschiedene XOR-Verknüpfungen bildet, ist anzunehmen, dass die Ergebnisse für weitere Arbeitsschritte verwendet werden. Polymorphe Malware hingegen führt die XOR-Verknüpfungen aus, um auf den Scanner wie ein sauberes Programm zu wirken. Allerdings werden im Anschluss die Ergebnisse verworfen, da sie nicht benötigt werden. Genau nach solchem auffälligen Verhalten sucht ein heuristischer Scanner.

Wird ein regelbasierter dynamisch-heuristischer Scanner eingesetzt, muss diesem zuerst ein auffälliges Verhalten von polymorpher Malware beigebracht werden. Dazu werden Regeln, die ein virales Verhalten beschreiben, definiert und geladen. Wird eine dieser Regeln erfüllt, erhöht oder senkt sich die Wahrscheinlichkeit dass es sich um infizierte Software handelt [5].

4.6 Metamorphe Malware

Metamorphose heißt Verwandlung. Metamorphe Malware kann ihre Reproduktionen auf syntaktische bzw. semantische Art und Weise modifizieren. Der Unterschied zwischen polymorpher und metamorpher gefährlicher Software liegt darin, dass erstere nur verschiedene Entschlüsselungsroutinen und unterschiedliche Schlüssel erstellen kann, metamorphe Malware hingegen in der Lage ist, ihren gesamten Code bei jeder Infektion variabel zu erzeugen. Daher besitzt sie im Gegensatz zu polymorpher Malware keinen konstanten Codebereich.

Auch Malware, die in der Lage ist, ausführbare Dateien auf verschiedenen Betriebssystemen (z.B. Linux und Windows) oder aber auf verschiedenen Computerarchitekturen zu infizieren, wird als metamorph bezeichnet. Erreicht wird dieses Ziel

zum Beispiel, indem die gefährliche Software verschiedene Portierungen als Codevarianten mit sich führt.

Metamorphe Malware ist aber nicht nur auf syntaktische Änderungen begrenzt. Einige Arten sind auch in der Lage, die Semantik ihres Codes zu ändern. Dazu gehört zum Beispiel das Ändern der Timing-Bedingung oder das Ändern der Nutzlast.

Es ist nicht notwendig, dass alle Reproduktionen metamorpher Malware funktionieren. Semantische Änderungen bösartiger Software können einen gewissen Umfang von Code-Korruption riskieren, wenn garantiert wird, dass ein bedeutsamer Anteil von allen Reproduktionen funktioniert und somit noch in der Lage ist, weitere funktionierende Reproduktionen zu erstellen.

Ein Ansatz zur Bekämpfung von metamorpher Malware ist die geometrische Entdeckung. Sie untersucht durch die Malware vorgenommene Änderungen an der Dateistruktur einer eventuell infizierten Datei. Diese Art der heuristischen Erkennung ist noch weit von einer hohen Erkennungsrate entfernt und neigt zu einer Vielzahl von Fehlalarmen [3].

Ein weiterer Ansatz metamorphe Malware zu entdecken ist die Code-Vereinfachung. Bei diesem Ansatz der Erkennung wird die verformte Malware von einer „höheren Schicht“ betrachtet. So setzt sich z.B. ein API-Aufruf aus einer bestimmten Anzahl an Programminstruktionen zusammen, die dann wiederum den eigentlichen API-Aufruf durch einen Interrupt auslösen. Die Instruktionen werden aus dem Befehlssatz der jeweiligen CPU zusammengesetzt. Durch die Vielzahl an Instruktionen und den Einsatz von Befehls-Substitution ist es daher möglich, für eine „höhere Aktion“, wie einen API-Aufruf, viele verschiedene Instruktionen zu verwenden.

Werden die verschiedenen Instruktionen der gefährlichen Software nicht betrachtet, sondern nur der dadurch eigentlich ausgelöste Befehl, so ist es möglich, jede metamorphe Malware zu erkennen, wenn einmal das eigentliche Verhalten bekannt ist. In diesem Fall ist es irrelevant, ob die metamorphe, gefährliche Software andere Verschleierungsmechanismen einsetzt. Dieser Ansatz der Erkennung kann somit auch bei verschlüsselter und polymorpher Malware angewandt werden.

Allerdings kann mit jedem Grad der Abstraktion bzw. mit jeder „höheren Aktion“ die Anzahl der Fehlalarme steigen. Eine Regel, die das Öffnen einer Datei mit Schreibzugriff als virales Verhalten bezeichnet, führt zum Beispiel dazu, dass bei Programm-Updates eine Warnung ausgeben wird, da auch hier Dateien mit Schreibzugriff geöffnet werden müssen. Auch der Einsatz von gefährlicher Software, die semantische Änderungen bei jeder Infektion vornimmt, kann diesem Verfahren Probleme bereiten.

Bei der Erkennung von Malware wird oft nach eindeutigen, identifizierbaren viralen Merkmalen gesucht. Eine weitere Möglichkeit, die Analyse von nicht vertrauenswürdigen Anwendungen zu beschleunigen ist es, genau nach dem Gegenteil zu suchen. So kann die Erkennungssoftware nach Verhalten bzw. Instruktionen suchen, die in keiner bekannten Form der zu suchenden metamorphen, gefährlichen Software vorhanden ist. Wird ein derartiges Muster entdeckt, kann der Scanner die Untersuchung abbrechen.

Fazit

Immer bessere Verschleierungsverfahren und eine stetig wachsende kommerzielle Nutzung des Internet, welche dieses auch in Zukunft als Ziel für Angriffe interessant macht, lassen noch komplexere und trickreicher verschleierte Malware erwarten.

Nach dem heutigen Entwicklungsstand gibt es noch keine Erkennungssoftware, die in der Lage wäre, alle unterschiedlichen Arten verschleierter Malware erfolgreich zu entdecken. Obwohl sich die heuristische Analyse als erfolgreiche Unterstützung für die signaturbasierte Virensuche etabliert hat, wird sie diese wegen ihrer Komplexität und Unschärfe nicht ersetzen. Allerdings ist zu erwarten, dass besonders die verhaltensanalytischen Fähigkeiten der heuristischen Analyse bei der Suche nach verschleierten Viren, Würmern und Trojanern eine stärkere Bedeutung erhalten werden.

Malware wird immer trickreicher und die eingesetzten Verschleierungsmechanismen werden von Generation zu Generation komplexer. Das gilt aber auch für die Erkennungssoftware. Daher gibt es bis heute (zum Glück) keine Malware mit einer „formellen Nicht-Erkennbarkeits-Garantie“.

Literatur

[1] Slade, Robert: Software Forensics. McGraw-Hill Education, 1 Auflage, 2004.

[2] Bania, Piotr: Fighting EPO Viruses. http://www.securityfocus.com/print/infocus/1841, Jun. 2005.

[3] Ször, Peter: The Art Of Computer Virus Research And Defense, Addison-Wesley, 1 Auflage, Feb. 2005.

[4] Adam Young, Moti Yung: Malicious Cryptography. Exposing Cryptovirology. John Wiley & Sons, 1 Auflage, Feb. 2004.

[5] Symantec: Understanding and Managing Polymorphic Viruses. Symantec White Paper Series, 30, Jul. 1999.

[6] BSI: http://www.bsi.de/av/vb/blaster.htm

[7] BSI: http://www.bsi.de/av/vb/tremor.htm