Fachhochschule Wedel
Seminar IT-Sicherheit
Sommersemester 2012
Dr. Gerd Beuster

Thema: Rootkits

Tobias Meyn

Seminar-Arbeit

1. Juli 2012

Abbildungsverzeichnis

1 Vorwort

Wenn man mit anderen Menschen über das Thema Computersicherheit diskutiert, so stößt man unweigerlich auf die Themengebiete „Hacker“, „Viren“ oder „Trojaner“. Sie umschreiben einen Bereich, in dem es um Fremdübernahme eines Systems oder dem Ausspionieren von Netzwerken geht. Ein Begriff, der nur selten genannt wird, ist der Begriff des „Rootkits“. Dies liegt nicht an der Tatsache, dass sie eine weniger große Gefahr darstellen, ganz im Gegenteil, sondern darin, dass vielen Rootkits nicht bekannt oder sich ihrer Funktionalität nicht bewusst sind.

Wir haben Virenscanner und IT-Sicherheitsprogramme, die uns melden, wenn wir Dateien auf dem System haben, in denen Trojaner oder ein Computervirus stecken. Was jedoch wäre, wenn wir keine Meldung bekommen? Wenn die Software, die von den Hersteller als Programm für mehr Computersicherheit angepriesen wird, gar nicht mehr das tun kann wozu sie programmiert wurde, da ein Rootkit auf gleicher Ebene Dateien und Kernel-Strukturen so umgebogen hat, dass der System-User in einer falschen Sicherheit gewogen wird? In einem Buch über Rootkits, schrieb der Autor von IT-Sicherheitsliteratur, Gary McGraw, in der Anpreisung des Buches *Rootkits – Subverting the Windows Kernel* [GB, 2006]:

„What´s worse than being owned? Not knowing it.”

In dieser Seminararbeit soll versucht werden, Rootkits zu erklären und auf die Gefahr, die von ihnen ausgeht, hinzuweisen. Desweiteren soll, anhand von vergangenen und gegenwärtigen Fällen von Rootkit-Nutzung, gezeigt werden, wie aktuell und präsent das Thema ist. Abschließend beschäftigt sich diese Ausarbeitung mit den Mitteln und Wegen, Anzeichen für Rootkits zu erkennen, sie zu identifizieren und zu entfernen.

2 Grundlagen von Rootkits

Um zu verstehen, was ein Rootkit ist und welchen Nutzen es hat, muss man verstehen, wie ein Angriff grundsätzlich abläuft und welche Absichten und Ziele ein Angreifer, im Folgenden auch Cracker genannt, verfolgen kann.

2.1 Ablauf eines Angriffs

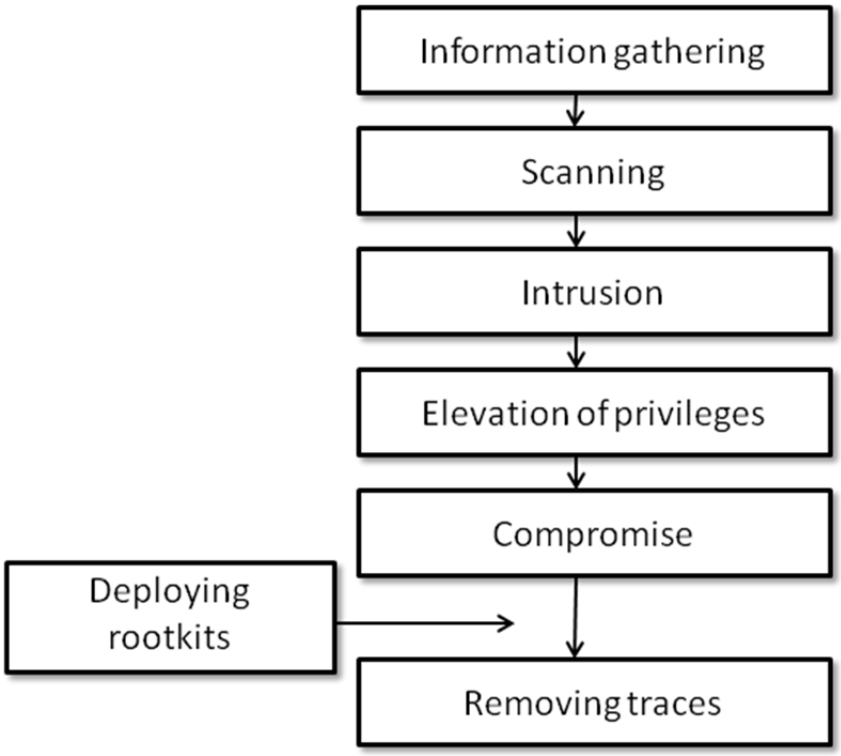

Abbildung 1 – Stages of an Attack[EM, 2011]

Der Cracker untersucht das Zielsystem und beschafft sich Informationen. So kann sich der Angreifer, bspw. mit Hilfe eines Portscans, die auf dem System laufenden Dienste anzeigen lassen. Die erlangten Informationen werden genutzt um eine Schwachstelle ausfindig zu machen. Mit Hilfe eines Exploits[1] werden die aufgedeckten Sicherheitslücken zum Eindringen in das System genutzt.

[1] **Exploit:** […] eine systematische Möglichkeit, die bei der Entwicklung eines Programms nicht berücksichtigt wurde, und wobei mit Hilfe von Befehlsfolgen Sicherheitslücken und Fehlfunktionen von Programmen ausgenutzt werden.

Der nächste Schritt eines Angriffs ist das Beschaffen von Privilegien, die, wenn nicht schon über das Exploit, über lokale Schwachstellen erlangt werden. Nachdem der Cracker nun Zugang zu allen Systembereichen hat und als Administrator oder, wie in Unix-Betriebssystemen üblich, Root-User arbeiten kann, kompromittiert[2] er das System und modifiziert es so, dass es seine Ziele umsetzt.

2.2 Motivation eines Angreifers

Die Motivation eines Angreifers in ein anderes System einzudringen kann unterschiedlicher Natur sein: so lassen sich Daten aus dem System exfiltrieren, wie z.B. Passwörter, Kreditkartennummern. Eine weitere Möglichkeit, die ein Angreifer nach der Übernahme eines System hat, ist das "Lauschen" um beispielweise Tastendrücke des Systems aufzeichnen oder Netzwerkverbindungen zu speichern. Desweiteren kann der Angreifer eine Zerstörung des Systems beabsichtigen. Bei den Möglichkeiten, die sich dem Angreifer bieten, benötigt er Unsichtbarkeit (engl. stealth) um zu verhindern, dass das kompromittierte System von der Anwesenheit eines "Fremden" etwas mitbekommt, da er sich sonst gefährden würde.

[2] **Kompromittierung**: Ein System wird als kompromittiert betrachtet, wenn Daten manipuliert sein könnten und wenn der Eigentümer bzw. Administrator des Systems keine Kontrolle über die korrekte Funktionsweise oder den korrekten Inhalt mehr hat.

2.3 Was ist ein Rootkit?

Bei einem Rootkit handelt es sich um eine Sammlung von Softwarewerkzeugen, ähnlich eines Werkzeugkastens (engl. "kit"), die auf einem kompromittierten System installiert werden, um dem Angreifer dauerhaften Zugriff und Rechte des Administrators (in UNIX/Linux "Root") zu gewährleisten. Die wesentliche Möglichkeit, die einem ein Rootkit bringt, ist Unsichtbarkeit.

Gängige Funktionen von Rootkits:

- Dateien und Ordner verstecken
- Fernzugriff
- "lauschen"

2.4 Was ein Rootkit *nicht* ist

Ein Rootkit ist kein Exploit

Ein Rootkit lässt sich zwar in Verbindung mit einem Exploit nutzen, jedoch hat ein Rootkit an sich keine "ausnutzenden" Absichten, wie es ein Exploit hat. Nachdem ein Exploit vonstatten gegangen ist, wird ein Rootkit typischerweise installiert.

Ein Rootkit ist kein Virus

Ein Virus ist ein sich selbst ausbreitender Automaton[3]. Im Vergleich dazu ist ein Rootkit in keiner Weise selbstständig oder hat den Sinn, sich zu vervielfältigen sondern ist eine, von einem Angreifer gesteuerte, Funktion, die eine komplette Kontrolle des Systems ermöglicht.

[3] entlehnt aus dem lateinischen Adjektiv **automatus** (dt.: aus eigenem Antrieb handelnd)

3 Daten/Fakten

3.1 Ursprung

Das Konzept von Rootkits ist nicht generell neu. Viele der Methoden, die in modernen Rootkits verwendet werden, sind dieselben die auch in den Computerviren um 1980 herum verwendet wurden – z.B. die Modifikation von Key-System-Tables oder Programm-Logik. In dieser Zeit nutzten Computerviren noch Floppy Disks zum Verbreiten der infizierten Programme. Durch die Einführung von Windows NT durch Microsoft veränderte sich das Speichermodell und Programmierer von Computerviren blieben dem Windows Kernel fern.

In den frühen 90ern fanden viele Hacker heraus, wie sie sich Zugang zu einem System verschaffen konnten - mit Hilfe eines „buffer overflows“ [4]. Der Angreifer drang ins System ein, kompromittierte es und nutze das System um weitere Angriffe zu starten. Nachdem der Angreifer ins System gelang, musste er den Zugang aufrechterhalten – aus dieser Tatsache entstand das erste Rootkit.

Die ersten Rootkits waren zumeist lediglich „backdoor“-Programme, die nur wenig Tarnung nutzten. So wurde beispielsweise das Programm „ls“ - listet alle Dateien und Verzeichnisse auf - nur so umgeschrieben, das es bspw. den Ordner „hacker_data“ nicht anzeigt. So wurden dem User die Daten des Hackers gegenüber verborgen. Die Reaktion von System Administratoren war es, Programme zu entwickeln, die testeten ob Dateien verändert wurden.

Der nächste Schritt der Hacker war es in den Kernel vorzudringen, um so sämtliche Sicherheitssoftware zu untergraben und den Kernel zu modifizieren. Die Modifizierung des Kernels machte nun sämtliche für den Angreifer benötigte Tarnung möglich. Diese Technik entspricht denselben Techniken, die in den frühen 80er-Jahren genutzt wurde, um Computerviren vor Anti-Viren- Software zu verstecken.

---

[4] **Pufferüberläufe** (engl. ***buffer overflow***) gehören zu den häufigsten Sicherheitslücken in aktueller Software, die sich u. a. über das Internet ausnutzen lassen können. Im Wesentlichen werden bei einem Pufferüberlauf durch Fehler im Programm zu große Datenmengen in einen dafür zu kleinen reservierten Speicherbereich, den *Puffer*, geschrieben, wodurch nach dem Ziel-Speicherbereich liegende Speicherstellen überschrieben werden.

3.2 Aktuelle Beispiele

Rootkits sind nicht nur Theorie in der Hacker-Szene, sondern finden sich bisweilen schon im Alltag wieder. Im Folgenden sind zwei Beispiele beschrieben, die in nicht allzu ferner Vergangenheit liegen bzw. deren Folgen noch andauern (Stuxnet).

3.2.1 Sony´s Extended Copy Protection

2005 entdeckte der Sicherheitsexperte Mark Russinovich zufällig ein Rootkit auf, durch Digital Rights Management (DRM) kopiergeschützten, CDs von Sony und BMG. Es installierte sich automatisch, wenn die, auf der CD befindliche, Abspiel-Software für PCs gestartet wird. Zwar muss einer Lizenzvereinbarung zugestimmt werden, jedoch stand in ihr nichts davon, dass eine Software installiert werden würde oder Funktionen dem Nutzer verheimlicht werden.

Der Kopierschutz fragte im 2 Sekundentakt alle laufenden Prozesse nach geöffneten Datien ab, um unerwünschte Kopien zu verhindern. Diese Funktion führte das Rootkit sogar aus, wenn die CD nicht mehr im Laufwerk lag.

Desweiteren versteckte die Software nicht nur die ihr zugehörigen Dateien, Verzeichnisse, Prozesse und Registry-Schlüssel, sondern global alles, was mit *sys* im Namen anfängt. So ergab sich für Angreifer die Möglichkeit, Schadsoftware durch entsprechende Namensgebung mit Hilfe des Kopierschutzes zu tarnen.

Nach mehrfachen Klagen gegen Sony, entschloss sich der Hersteller zu einem Update, welches die Rootkit-Komponente entfernt. Um dieses zu installieren, mussten sich Nutzer jedoch zuerst auf der Website von Sony registrieren. Zusätzlich ergaben Analysen, dass die Deinstallation nur die Funktion zum Verstecken von Dateien/Verzeichnissen mit sys entfernte, den Kopierschutz aber bestehen lies.

3.2.2 Stuxnet

Stuxnet ist die Bezeichnung für einen Computerwurm, der im Juni 2010 mehrheitlich im Iran entdeckt wurde. Das Schadprogramm wurde speziell für ein Siemens-System entwickelt und ließ eine Debatte entbrennen, ob Stuxnet von den USA und Israel entwickelt wurde um das iranische Atomprogramm zu stören.

CYBERWAR

Obama befahl Stuxnet-Attacken gegen Iran

Die Cyber-Angriffe auf iranische Atomanlagen von 2010 gingen von den USA aus. Ein Reporter der New York Times beschreibt in einem Buch, wie Obama das Programm vorantrieb.

© Jewel Samad/AFP/GettyImages

US-Präsident Barack Obama

US-Präsident Barack Obama hat einem Bericht der *New York Times* zufolge Cyberattacken gegen den Iran angeordnet, bei denen die Regierungsexperten auch den berüchtigten Computerwurm Stuxnet einsetzten. Als Stuxnet im Sommer 2010 öffentlich bekannt wurde, habe Obama die Geheimaktion mit dem Codenamen *Olympic Games* noch

Abbildung 2 – Artikel der ZEIT über Stuxnet [ZEIT, 2012]

So schrieb die die New York Times, dass Stuxnet es auf einen Verbund aus genau 984 Maschinen abgesehen hatte – und genau ein solcher wurde im Sommer 2009 im Iran stillgelegt, wie Atom-Experten berichteten. Das besondere an Stuxnet ist die Installation eines PLC[5]-Rootkits zur Modifikation der Steuerungssysteme. [NYT, 2011]

[5] Ein **Programmable Logic Controller** *(dt.* **speicherprogrammierbare Steuerung)** ist ein Gerät, das zur Steuerung- oder Regelung einer Maschine oder Anlage eingesetzt wird und auf digitaler Basis programmiert wird.

3.3 Trend

Die nachfolgenden Statistiken sind einem White Paper von McAfee® entnommen und das Erscheinungsdatum liegt im Jahr 2006, dennoch gibt die Aufzeichnung einen Einblick auf die Entwicklung und den Anstieg von Rootkit-Technologie. [McA, 2006/2007]

3.3.1 Verbreitung

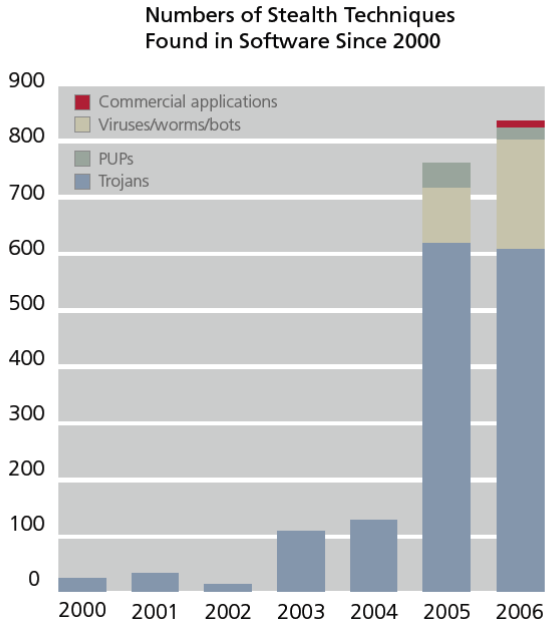

Abbildung 3 – Verbreitung von Rootkits

In den Jahren 2003 bis 2006 hat sich das Vorkommen von Stealth-Technologien in Malware, PUPs und kommerziellen Anwendungen mehr als versechsfacht. Wie Abbildung 3 zeigt, war der Einsatz von Stealth-Technologien in 2005 nicht länger die alleinige Domäne von Trojanern, sondern zeigte sich auch in anderen Formen von Malware, PUPs[6] und kommerziellen Anwendungen.

Über Internetseiten und Webblogs verbreitet sich der Quellcode von Rootkits und einfache Programme bieten selbst Laien Stealth-Techniken an. So gibt es Stealth-Creation Kits, wie die Anwenderschnittstelle des *Nuclear Rootkit: sie* benötigt den Namen einer Datei oder eines

[6] **PUP**s (potenziell unerwünschte Programme) sind Programme, die mit dem stillschweigenden Einverständnis des Anwenders installiert und ausgeführt werden.

Verzeichnisses, um dann mit einem Mausklick mehrere Stealth-Techniken einzusetzen. So wird ein maßgeschneiderter Binärcode entworfen, der die Datei oder das Verzeichnis versteckt. Diese Online-Zusammenarbeit von Hackern und die einfache Verfügbarkeit stellen eine erhebliche Herausforderung für die Sicherheitsgemeinschaft dar, da die zunehmende Weiterentwicklung es immer schwieriger macht, Rootkits zu verhindern, zu entdecken und zu entfernen

3.3.2 Komplexität

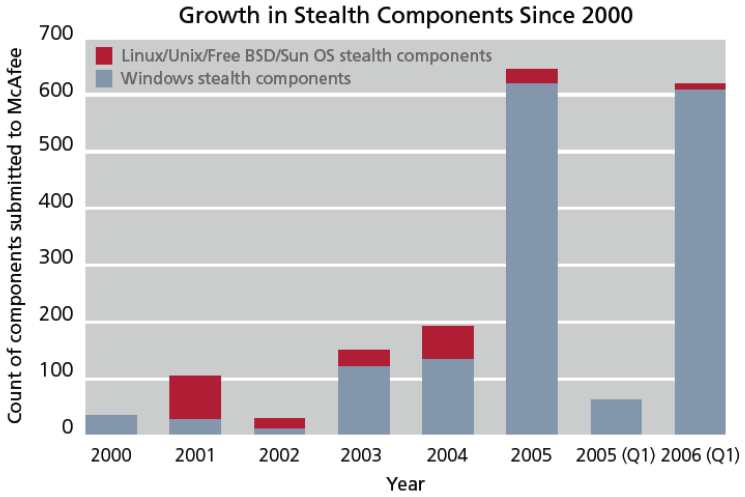

Abbildung 4 – Komplexität von Rootkits

Die Zusammenarbeit von Hackern in Internetforen führte zu einer immer größeren Komplexität der Rootkits. Programmierer dieser Technologien finden in der Windows-Plattform, bedingt durch viele undokumentierte APIs, eine technische Herausforderung. Diese Tatsache, in Verbindung mit der verbreiteten Installation, begründen die kontinuierlich steigende Komplexität von Windows Stealth Components (siehe Abbildung).

4 Arten von Rootkits

4.1 User-Level-Rootkits

User-Mode-Rootkits laufen auf User-Ebene und modifizieren Dateien und Kommandos auf einem Computer mit administrativen Rechten. Sie werden unterschieden in Binär- und Library-Rootkits und stellen die am einfachsten aufzuspürende Art von Rootkits dar.

4.1.1 Binary-Rootkits

Binäre Rootkits haben ihren Namen daher, dass sie die "binaries" von ausführbaren Systembefehlen verändern, um so zusätzliche Funktionen zu bieten oder andere in gewünschter Weise zu unterlassen (Anzeige von Ports o.ä.). So können beispielsweise die Befehle "ls" (Unix) und "dir" (Windows) so modifiziert werden, dass sie Objekte, die einem, vom Angreifer definierten, String entsprechen, nicht anzeigen. Eine weitere Anwendungsmöglichkeit ist die Veränderung des Befehls "netstat" um aktive Ports auszublenden, "ps" um Prozesse zu vestecken oder im "ssh" ein Login einzubauen, den der Rootkitersteller nutzen kann.

4.1.2 Library-Rootkits

Ein Library-Rootkit nutzt den vordefinierten, zur Laufzeit eines Systemprogramms, geladenen Code. Dieser Code wird aus dynamischen Bibliotheken in den Anwendungsspeicher geladen und verlinkt. (Windows: **D**ynamic **L**inked **L**ibrarys - .DLL, Unix: **S**hared **O**bject - .SO) So kann ein Programm beispielweise eine Windows-Schnittstelle nutzen, die ihre Daten aus der beispiel.DLL bezieht. Ein User-Rootkit würde diese Schnittstelle so modifizieren, dass sie stattdessen auf den Rookit-Code zugreift. Der Rootkit-Code wird dann üblicherweise auf die beispiel.DLL zugreifen und die Ausgabe modifizieren. Ein unter Unix bekanntes Library-Rootkit ist "t0rn", welches die Systembibliothek libproc.so austauscht. Diese hat ihren Nutzen in der Bereitsstellung von Prozessinformationen über das /proc Dateisystem. Durch die Modifikation der *.so-Datei werden bestimmte Prozesse nicht mehr angezeigt, da die Informationen nicht an ein entsprechendes Kommando gesendet werden.

4.2 Kernel-Level-Rootkits

Kernel-Level-Rookits zielen auf den Betriebssystemkern, also den zentralen Bestandteil des Betriebssystems, ab. Der Kernel-Level stellt die unterste Schicht und somit die direkte Anbindung zur Hardware bei voller Befehlsgewalt dar: der Administratorebene. Ein KL-Rookit bietet dem Angreifer die Möglichkeit, seine Absichten auf dieser Ebene auszuführen. Durch die Tatsache, dass das Rootkit auf demselben Level arbeitet, auf dem auch das Betriebssystem und potenzielle Antivirus-Programme laufen, stellt diese Ebene für das Rootkit die „sicherste“ dar. Wird die Installation von Gerätetreibern unterstützt, lassen sich Rootkits - als Gerätetreiber (Windows) oder Kernmodule (Linux) geschrieben - auf Kernel-Ebene installieren. Unter Windows ist es einem Rootkit möglich, Datenstrukturen im Systemkern mit Hilfe einer Methode direkt zu modifizieren. Diese "**d**irect **k**ernel **o**bject **m**odification" (DKOM) kann Einträge in der **System Service Dispatch Table** (SSDT) – einer Kerneltabelle mit Pointern zu Funktionen und Methoden – verändern, um auf den Rootkit-Code zu verweisen und somit seine Anwesenheit verschleiern. In Linux ist dieser Vorgang mit der Veränderung der System Call Table – einer Liste aller bekannten Systemaufrufe innerhalb des Linux-Kernels – ähnlich möglich.

Die erhöhte Schwierigkeit beim Schreiben von Kernel-Level-Rootkits machen Bugs in der Funktionalität zu keiner Seltenheit und wirken sich auf die Stabilität des betroffenen Systems aus. So sind unerwartete Blue-Screens und Systeminstabilität ein Indiz für Kernel-Level-Rootkits. Werden Kernel-Level-Rootkits mit User-Level-Rootkits kombiniert, spricht man von **Hybrid-Rootkits**.

4.3 Firmware-Rootkits

Firmware-Rootkits haben ihren Namen aus der Eigenschaft, dass sie ein dauerhaftes Abbild in der Hardware erstellen. So können z.B. Netzwerk-Karten, Festplatten oder das System-BIOS genutzt werden um das Rootkit zu verstecken. John Heasmen beschrieb die Überlebensfähigkeit von Firmware-Rootkits in seinem Report „Implementing and Detecting an ACPI BIOS Rootkit“. **[JH, 2007]**

Im März 2009 veröffentlichten die Forscher Alfredo Ortega und Anibal Sacco Details eines Windows-Rootkits, welches in der Lage war mit Hilfe des BIOS die Formatierung einer Festplatte zu überleben. **[AO, 2009]**

4.4 Virtuelle Rootkits

Virtuelle Rootkits verhalten sich nach dem Prinzip eines Hypervisors von Virtualisierungssoftware. Sie können nur Prozessoren mit Hardware-Virtualisierungs-Unterstützung befallen. Diese Prozessoren erlauben es, Kernel-Mode-Software bei bestimmten Prozessorbefehlen abzufangen, beispielsweise bei I/O-Befehlen. The Blue Pill ist ein solches virtuelles Rootkit. Dabei handelt es sich um eine Technologiedemonstration der Hackerin Joanna Rutkowska, die sie auf der Black-Hat Konferenz 2007 vorstellte. Das Blue Pill-Konzept ist es, eine laufende Instanz des Betriebssystems mit Hilfe des Hypervisor zu virtualisieren. Das vorherige Betriebssystem würde immer noch ihre bestehenden Verweise auf die Hardware und Dateien besitzen, aber fast alles, einschließlich Hardware-Interrupts und Anfragen nach Daten, kann durch den Hypervisor abgefangen werden.

Der Ansatz des virtuellen Rootkit wird aber, durch die sich weiterentwickelnde Technologie, zunehmend Ziel von Rootkit-Programmierern sein, da sie nahezu unsichtbar sind.

5 Gegenmaßnahmen

Der wichtigste allgemeine Ansatz, das Betriebssystem vor Rootkits zu schützen, ist es, das System vom unbefugten Zugriff von Fremden zu schützen um so die Installation eines Rootkits für Angreifer gar nicht erst möglich zu machen.

5.1 Anzeichen für Rootkits

- Blue-Screens / System-Abstürze
- Auffällig hohe Netzlast
- Einfache Aktionen benötigen länger beim Aufruf
- Einstellungen im System ändern sich (ohne Grund)

5.2 Rootkits erkennen

Das Problem bei dem Aufspüren von Rootkits ist, dass zur Detektion ein Verdacht notwendig ist. Da ihre Kernfunktion im Tarnen von Funktionen und Prozessen liegt, können anhand von Anzeichen installierte Rootkit nicht mit Sicherheit identifiziert werden. Viele der modernen Rootkits sind programmiert um einfachen Detektion-Versuchen standzuhalten.

Erkennung und Entfernung hängen von den Verfälschungsqualitäten des Rootkits ab – je besser das Rootkit geschrieben ist, desto schwieriger ist es, alle infizierten Dateien zu erkennen.

5.2.1 Anti-Rootkit Programme

Da Anti-Rootkit Programme auf Kernel-Ebene laufen, ist es für User-Mode-Rootkits meist schon ausreichend, sogenannte Anti-Rootkit Programme laufen zu lassen um unerwünschte Kits aufzudecken. Nachfolgend sind drei solcher Programme aufgelistet und beschrieben.

5.2.1.1 RootkitRevealer

Das kostenlose Sicherheits-Tool "RootkitRevealer" spürt versteckte Rootkits auf, indem es die Registry absucht. Das Tool vergleicht anschließend die Scan-Ergebnisse auf höchstem und niedrigstem Gefahren-Level. Die höchste Ebene ist hierbei die Windows-API, die niedrigste Ebene ist der Rohinhalt eines Dateivolumens oder einer Registrierungsstruktur. Nach abgeschlossenem Scan-Durchlauf listet RootkitRevealer die Ergebnisse sortiert nach Pfad, Zeitstemple, Größe und Beschreibung auf.

5.2.1.2 Rootkit Hunter (Linux)

Rkhunter (Rootkit Hunter) ist ein Unix-basiertes Tool, das das System auf Rootkit, Backdoors und mögliche lokale Exploits scannt. Dies tut es, indem es „SHA-1 Hashes“[7] von wichtigen Dateien mit bekannten Strukturen in Online-Datenbanken vergleicht, wobei es nach Standard Verzeichnissen (von Rootkits), falschen Privilegien, versteckten Dateien und verdächtigen Strings in Kernel-Modulen sucht.

```
Datei  Bearbeiten  Ansicht  Terminal  Hilfe
    Trojanit Kit                                          [ Not found ]
    Tuxtendo Rootkit                                      [ Not found ]
    URK Rootkit                                           [ Not found ]
    Vampire Rootkit                                       [ Not found ]
    VcKit Rootkit                                         [ Not found ]
    Volc Rootkit                                          [ Not found ]
    Xzibit Rootkit                                        [ Not found ]
    X-Org SunOS Rootkit                                   [ Not found ]
    zaRwT.KiT Rootkit                                     [ Not found ]
    ZK Rootkit                                            [ Not found ]

  Performing additional rootkit checks
    Suckit Rookit additional checks                       [ OK ]
    Checking for possible rootkit files and directories   [ None found ]
    Checking for possible rootkit strings                 [ None found ]

  Performing malware checks
    Checking running processes for suspicious files       [ None found ]
    Checking for login backdoors                          [ None found ]
    Checking for suspicious directories                   [ None found ]
    Checking for sniffer log files                        [ None found ]

  Performing Linux specific checks
    Checking loaded kernel modules                        [ OK ]
    Checking kernel module names                          [ OK ]

[Press <ENTER> to continue]
```

Abbildung 4 – Rootkit Hunter Anwendung

5.2.1.3 Microsoft-Tool zum Entfernen bösartiger Software

Das Microsoft-Tool überprüft Computer, auf denen Windows 7, Windows Vista, Windows XP, Windows 2000 oder Windows Server 2003 ausgeführt wird, auf Infizierung durch spezifische, verbreitete Schadsoftware. Sobald der Vorgang zur Erkennung und Entfernung entsprechender Infizierungen abgeschlossen ist, zeigt das Tool die Ergebnisse in einem Bericht an, aus dem zu erkennen ist, welche bösartige Software ggf. erkannt und entfernt wurde.

---

[7] **SHA**, bezeichnet eine Gruppe standardisierter, kryptologischer Hashfunktionen. Diese dienen zur Berechnung eines eindeutigen Prüfwerts (Digitale Signatur) für beliebige digitale Daten (Nachrichten).

5.2.1.4 Funktionsweise

Die beschriebenen Programme arbeiten mit Signaturen und Mustern von bekannten - sogenannten *„known-good“* - Technologien. Der Nachteil an dieser Methode ist, dass neuartige Rootkittechnologien, die noch unbekannt sind, zumeist nicht erkannt werden.

Zusätzlich stellt die beschriebene Tatsache, dass diese Programme auf Kernel Ebene laufen, gleichwohl auch das Problem dieser Programme, was das Entdecken von Kernel-Level-Rootkits angeht, dar. Zwar können einige Rootkits vielleicht aufgedeckt werden (bspw. bei Hybrid-Rootkits), jedoch kann der User nicht sicher sein, dass das System danach von allen Rootkits gesäubert wurde. Auf Kernel-Ebene könnten Strukturen verändert worden sein, die Anti-Rootkit-Programme nutzen.

5.2.2 Boot CD

Ein weiterer Ansatz ist es, eine Boot-CD in Verbindung mit einem Malware-Scanner zu nutzen um die Rootkits auf dem System zu entdecken. Das Konzept, einen solchen Scanner von einer Boot-CD zu starten, stellt sicher, dass die Betriebssystemumgebung des Scanners nicht verseucht sein kann. Ein Schädling kann so weder den Scanner selbst manipulieren, noch sich mit Funktionalitäten eines Rootkits unsichtbar machen, da sein Programmcode nicht während des Systemstarts von der Festplatte ausgeführt wurde.

5.2.3 VM Technologie (Paladin)

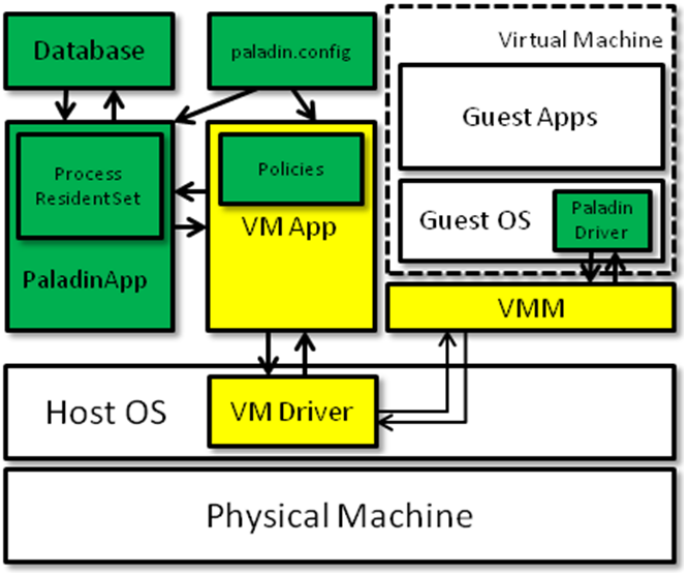

Abbildung 5 – Architektur von Paladin

VM-Technologie kann ebenfalls genutzt werden, um Kernel-Level Rootkits zu erkennen. Paladin ist ein Beispiel für ein Paket, das die virtuelle VMware-Maschine verwendet, um Rootkits zu erkennen. Die angewandte Methodik ist, Schutzzonen (geschützte Speicher- und Dateizonen) und den Zugriff auf diese definierten Zonen zu bewachen. So gehört das Speicherabbild des Kernels und verschiedene Jump-Tables[8] zu einer solchen bewachten Schutzzone. Zusätzlich werden Abhängigkeits-Bäume zwischen verschiedenen Betriebssystem-Objekten erstellt. Die Abbildung zeigt die Architektur von Paladin an. **[BCI, 2006]**

[8]**Sprungtabelle** (engl. **Jump Table**) ist eine Software-Konstruktion, um eine bestimmte Auswahl von Funktionen (aus einem Betriebssystem oder aus einer Funktionsbibliothek) bequem und kompatibilitätssicher aufrufen zu können.

5.3 Rootkits entfernen

Sobald ein Rootkit auf dem System identifiziert wurde, sind die Optionen zur Entfernung eingeschränkt. Zwar ist es möglich, mit Hilfe von Erkennungssoftware oder einigen Tools, bekannte Rootkits entfernen oder herausfinden, welche Dateien verändert wurden, jedoch geben diese Schritte keine Garantie für ein von Rootkits befreites System. Weil sich Rootkits verstecken können, kann man nicht wissen, wie lange es sich schon auf dem System befindet. Es ist auch nicht möglich zu wissen, welche Informationen und Strukturen kompromittiert wurden. Die beste Reaktion auf eine identifizierte Rootkit ist es daher das System zu formatieren und neu zu installieren. Diese drastische Methode ist die einzige bewährte Methode, um Rootkits vollständig zu entfernen.
Zu verhindern, dass Rootkits auf das System gelangen, ist die beste und wichtigste Strategie im Kampf gegen Rootkits. Dies kann mit denselben Maßnahmen geschehen, die auch im Kampf gegen andere Malware, wie Trojaner und Computerviren, angewendet werden. Hierzu gehören Virenscanner, regelmäßige Software-Updates, eine Firewall auf dem Host und dem Netzwerk, und eine sichere Kennwort-Strategie.

6 Ansatz: Trusted Computing

Ein in der Diskussion stehender Ansatz ist das sogenannte „Trusted Computing”. Mit „trusted“ wird gemeint, dass ein Gerät oder eine Software die Erwartung, dass es sich für einen bestimmten Zweck in einer, meist vom Hersteller, vordefinierten Art und Weise verhält. Im Rahmen des „Trusted Computing“, welches von der Trusted Computing Group (kurz: TCG) entwickelt und beworben wird, bedeutet es, dass der Betreiber eines PC-Systems die Kontrolle über die verwendete Hard- und Software an dritte abgeben kann. Dies dient, laut der Beschreibung der TCG, der Sicherheit.
Ein zusätzlich in der Hardware verankertes Trusted Platform Module ist das erste Glied einer Sicherheitskette („Chain of Trust“), die sich vom Beginn des Bootvorgangs bis zum Start der Applikationen erstreckt. Sobald jeweils die untere Ebene über eine stabile Sicherheitsreferenz und keinerlei infiziertem Code verfügt, kann die nächste Ebene darauf aufbauen. Jede dieser Ebenen baut auf der vorhergehenden auf und erwartet damit, dass im Gesamtsystem interne Verbindung und Geräteanbindung vertrauenswürdig, zuverlässig, sicher und geschützt ist.
Kritiker des Konzepts verweisen auf die Tatsache, dass die Lizensierung der verwendeten Produkte und die Urheberrechte von gespeicherten Inhalten jederzeit überprüft und an Dritte übermittelt werden könnten. Der Benutzer muss sein Vertrauen in die Sicherheitskomponenten auf Schlüssel begründen, die er nicht selbst erzeugt hat. Ob die gespeicherten Schlüssel wirklich geheim und einmalig sind, ist nicht überprüfbar. Außerdem besteht die Gefahr, dass Hersteller von Hardware und Betriebssystemen die neuen Sicherheitsinitiativen nutzen, um bestehende Monopole zu festigen oder neue aufzubauen. Restriktive Systeme könnten die Verwendung von Open-Source Alternativen zu den rein kommerziellen Lösungen stark einschränken oder gar zum Erliegen bringen. So illustrieren die Bestrebungen von Intel und Microsoft in diesem Bereich die Entwicklungen in den kommenden Jahren bei allen großen Herstellern.

7 Fazit

Rootkits stellen eine Technologie dar, dessen Bedeutung und Gefährlichkeit in der Computersicherheit nicht in Relation zu dem Wissen und dem Verständnis von ihnen stehen. Wie das Seminar aufgezeigt haben sollte, sind Rootkits vielfältig und eine stetig wachsende Bedrohung.

Eines der beunruhigenden Merkmale, ist die unter Trends (Siehe 3.3.1) beschriebene Verbreitung von Rootkits-Code. So bieten Internetforen und Webblogs Möglichkeiten einzelner Rootkit-Module immer weiter zu verfeinern. Die weitere Entwicklung deutet sich an dem Blue-Pill Rootkit und Paladin an: der virtuelle Ansatz in der Rootkit-Erstellung und -Bekämpfung wird sich weiterentwickeln und ermöglicht dem VM Layer, Kompromittierungen im Kernel erkennen zu können.

Die Reaktion der Hacker-Szene auf die Möglichkeit, User-Mode-Rootkits zu erkennen, indem auf Kernel-Ebene gearbeitet wurde und der daraus resultierenden Entwicklung von Kernel-Mode-Rootkits sind im Bereich der virtuellen Rootkits ebenfalls denkbar und stellen für die Hacker-Szene eine neue Herausforderung dar.

Kernpunkt dieses Seminars ist die Nachricht, dass das größte Problem nicht im Computer sondern "vor" dem Computer sitzt: mangelndes Wissen oder fehlende Sicherheitssoftware, die dem Angreifer die Möglichkeit geben, dass System zu kompromittieren, werden weiterhin Angriffsfläche bieten. Ist ein System von einem Rootkit befallen, so kann der User nicht wissen, wie viel seines Systems modifiziert wurde.

Quellenverzeichnis

[AC, 2003] **Anton Chuvakin** (2003)
An Overview of Unix Rootkits

[AB, 2004] **Andreas Bunten** (2004)
UNIX und Linux basierte Kernel Rootkits

[SY, 2005] **Symantec** (2005)
Windows Rootkit Overview

[MB, 2005] **Mike Danseglio, Tony Bailey** (2005)
Rootkits: The Obscure Hacker Attack

[LGO, 2005] **J.F. Levine, J.B. Grizzard, H.L. Owen** (2005)
Detecting and categorizing kernel-level rootkits to aid future detection

[GB, 2006] **Greg Hoglund, James Butler** (2006)
Rootkits – Subverting the Windows Kernel

[BCI, 2006] **Arati Baliga, Xiaoxin Chen, Liviu Iftode** (2006)
Paladin: Automated Detection and Containment of Rootkit Attacks

[McA, 2006/2007] **McAfee** (2006/2007)
Rootkits - The Growing Threat / A Technical Primer

[JH, 2007] **John Heasmen** (2007)
Implementing and Detecting a PCI Rootkit

[AO, 2009] **Anibal Sacco, Alfredo Ortéga** (2009)
Persistent BIOS infection

[EM, 2011] **Erez Metula** (2011)
Managed Code Rootkits – Hooking into Runtime Environments

[SY, 2011] **Symantec** (2011)
W32.Stuxnet Dossier

Internetlinks

[NYT, 2011]
http://www.nytimes.com/2011/01/16/world/middleeast/16stuxnet.html

[ZEIT, 2012]
http://www.zeit.de/politik/ausland/2012-06/obama-iran-stuxnet