DuD
Datenschutz und Datensicherheit

Vertrauenswürdige Protokollierung

Protokollierung mittels nicht-deterministischer nebenläufiger wechselseitiger Überwachung

Stephen D. Wolthusen

Die Etablierung von Mehrkern-Prozessoren als Standard-Ausstattung von Arbeitsplatzrechnern erlaubt es, neue Verfahren zur wechselseitigen nebenläufigen Überwachung von Protokollierungsinstanzen sowie der parallelen und nichtdeterministischen Erfassung von Invarianten zu entwickeln. Dies erlaubt die Erkennung von Kompromittierungsversuchen selbst in dem Fall, dass Angreifer die Verteidigungsmechanismen vollständig kennen.

Prof. Stephen Wolthusen

lehrt IT-Sicherheit am NISlab, Gjøvik University College, Norwegen sowie im Fachbereich Mathematik, Royal Holloway, University of London (UK).
E-Mail: stephen.wolthusen@rhul.ac.uk

Einleitung

Die Vertrauenswürdigkeit von Protokolldaten sowie deren Erfassungs- und Verarbeitungsschritte stellt neben deren revisionssicheren Weiterverarbeitung eine wesentliche Herausforderung dar. Dies ist nicht zuletzt dadurch begründet, dass eine unerkannte Kompromittierung durch Schadsoftware wie Trojanische Pferde und sogenannte Rootkits [3,6,7] nicht a priori ausgeschlossen werden kann.

1 Kompromittierung von Systemen und Protokolldaten

Eines der ersten Ziele eines Rootkits bzw. eines Trojanischen Pferdes ist es, durch Gegenmaßnahmen wie Intrusion Detection-Systeme nicht erkannt werden zu können. Anders als im Fall von Viren ist solche Schadsoftware auf möglichst langfristige Kompromittierung hin ausgelegt und zumeist auch deutlich umfangreicher, sodass hier unabhängig von der konkreten Zielsetzung (Nutzlast, z.B. zur Ausspähung, aber auch als sogenannter „Zombie“ für verteilte Systeme zum Versand von Spam und ähnlichen Vorgängen) erheblicher Aufwand zur Maskierung der Tätigkeiten und Kommunikations-Mechanismen betrieben wird.

Für die Protokollierung und ihre Vertrauenswürdigkeit bedingt dies jedoch gleichzeitig, dass eine Kompromittierung einerseits Abstreitbarkeit bedingt, da unbefugte Handlungen in diesem Fall als Bestandteil der Nutzlast der Schadsoftware, jedoch nicht länger dem Nutzer selbst zweifelsfrei zugeordnet werden können, andererseits aber insbesondere auch, dass die Protokolldaten selbst auf unbestimmte Zeit in der Vergangenheit infrage gestellt werden müssen. Letzteres ist dadurch bedingt, dass in Abwesenheit eines klaren Zeitpunktes, an dem eine Kompromittierung identifiziert werden kann, grundsätzlich alle Protokolldaten eines derart infizierten Systems als nicht vertrauenswürdig betrachtet werden müssen.

Neuere Rootkits wenden eine Reihe von Verfahren an, welche ihre Auffindung und Entfernung erschweren. So existieren Systeme, welche sich die persistente Speicherung außerhalb des normalen Betriebssystems bzw. von Massenspeichern nutzen, indem sie die ACPI-Konfigurationsspeicher verwenden [10]. Ein weiterer Ansatz, welcher Verteidigungsmechanismen weitgehend wirkungslos werden lässt, ist die Installation des Rootkits als Virtualisierungsumgebung, welche anschließend das kompromittierte System innerhalb einer virtuellen Umgebung unter Verwendung der Hardware-Virtualisierungsschicht moderner Prozessoren betreibt und somit die eigene Existenz vollständig vor dem Opfer-System verbirgt [15].

Da moderne Rootkits nach ihrer Einbettung in ein System dadurch nur noch bedingt auffindbar sind [7,8], ist es von entscheidender Bedeutung, den Zeitpunkt des Angriffs bzw. der (versuchten) Kompromittierung festzuhalten, selbst wenn der eigentliche Angriff dennoch erfolgreich ist.

2 Nebenläufige Messungen

Moderne Betriebssysteme stellen Anwendungsprogrammen und Betriebssystem-Funktionen eine simulierte Nebenläufigkeit in Form des sogenannten Präemptiven Multitaskings bereit, bei dem einzelne Prozesse jeweils Zeitscheiben der zentralen Ressource CPU zugeteilt bekommen und diese exklusiv für einige Millisekunden

bzw. bis zur freiwilligen Aufgabe der Zeitscheibe (z.B. bei Anforderung von Daten, die von Peripheriegeräten eingelesen werden müssen) nutzen. Die Reihenfolge der Einteilung von Zeitscheiben durch das Betriebssystem folgt hierbei zwar einem einfachen und vorhersehbaren Algorithmus, jedoch ist es einem Beobachter in einem Prozeß ohne genaue Kenntnis des Systemzustandes dennoch nicht ohne weiteres möglich, die exakte Reihe vorherzusagen, da externe Ereignisse wie Wartezeiten bei Gerätezugriffen und die freiwillige Aufgabe von Zeitscheiben durch andere Prozesse diese dynamisch verändern können.

Im konkreten Fall der Einbettung eines Rootkits kann unter diesen Voraussetzungen nicht ausgeschlossen werden, dass das Rootkit die zur eigenen Einbettung und anschließenden Maskierung seiner Präsenz erforderlichen Schritte durchführt, bevor ein Verteidigungsmechanismus überhaupt die Gelegenheit erhält, geeignete Messungen anzustellen. Dies gilt insbesondere für fortgeschrittene Techniken wie die sogenannte Cross-View-Analyse, bei der versucht wird, auf verschiedenen logischen Ebenen von Gerätetreibern bis hin zu Anwendungsschnittstellen auf ähnliche Daten (z.B. Größe und Modifikationsdaten von Dateien) zuzugreifen, da das Rootkit aufgrund der oben angeführten Betrachtung bereits in der Lage gewesen sein kann, diese Inkonsistenzen vollständig zu maskieren [8].

Nimmt man jedoch an, dass ein Rootkit zur Erzielung des gewünschten Effekts stets Folgen von mindestens zwei nicht-atomaren Operationen (d.h. zumindest eine Operation für die gewünschte Funktion sowie mindestens eine zur Maskierung) durchführt, eröffnet die Verfügbarkeit von Mehrkern-Prozessoren neue Verteidigungsmöglichkeiten in der Form der aktiven Ausnutzung von Nebenläufigkeit und Nichtdeterminismus.

Die zuvor genannten Cross-View-Verfahren beruhen darauf, dass ein bestimmter Systemzustand i.d.R. an mehreren Stellen im System, etwa auf verschiedenen Abstraktionsebenen reflektiert ist, und ein Rootkit auf einer oder mehreren Ebenen nach Einfügung seiner Nutzlast diese zur Maskierung abändern muß, um messbare Inkonsistenzen zu vermeiden. Dabei kann die Granularität dieser Operationen sehr feinkörnig bis hin zu atomaren Operationen sein, d.h. etwa lediglich die Veränderung einzelner Speicherplätze im Arbeitsspeicher. Während aufgrund der vorgenannten Struktur des Betriebssystems derartige Änderungen nicht zwingend erkannt werden können, da diese bereits vollständig abgeschlossen sind, bevor ein Detektionsmechanismus zum Tragen kommt, ist durch die Verfügbarkeit moderner Mehrkern-Prozessoren (sowie der bereits lange verfügbaren symmetrischen Multiprozessor-Systeme (SMP), welche jedoch außer im Server-Bereich nur begrenzte Verbreitung haben) eine weitere Möglichkeit gegeben.

Formal betrachtet stellen die Messungen der Cross-View-Verfahren eine wohldefinierte Relation zwischen Variablen her, im einfachsten Fall eine Invariante, wobei auch Ungleichungen und komplexere Relationen möglich sind. Darüberhinaus sind vielfach Invarianten und Relationen über mehr als zwei Variable anzutreffen, da moderne Betriebssysteme zumeist auf mehreren logisch aufeinanderbauenden Schichten basieren und so Zustände z.B. des Dateisystems, der Speicherverwaltung, oder auch des Netzwerk-Protokollstapels auf jeder dieser Abstraktionsebenen repräsentieren.

Eine Manipulation, insbesondere zur Maskierung von Schadsoftware, wird daher zu einer ggf. auch nur temporären Verletzung einer solchen Relation oder Invariante führen.

Führt man nun für die einzelnen Elemente der Relation bzw. Invariante Messungen parallel durch indem man jedes einzelne Element mindestens einem nebenläufigen Thread zuordnet und selbige Threads auf die verfügbaren Prozessorkerne verteilt, so läßt sich beobachten, dass selbst wenn nur zwei Prozessorkerne zur Verfügung stehen, eine qualitative Verbesserung gegenüber dem vorgenannten Sachverhalt eintritt. Dies ist dadurch bedingt, dass die Prozessorkerne echt nebenläufig sind, d.h. real und nicht nur in Außensicht parallel verarbeiten. Es ist weiterhin festzuhalten, dass in einer solchen Architektur (wie auch in stärkerem Maße bei SMP-Systemen) die einzelnen Prozessorkerne nicht synchron arbeiten, sondern aufgrund einer Vielzahl von Einflüssen wie z.B. Ressourcenkonkurrenz, Interrupts, dynamischer Anpassung der Taktfrequenz und selbst des internen Zustandes des Prozessorkernes, welcher eine unterschiedliche Anzahl von Taktzyklen für die Ausführung einer einzelnen Maschinen-Instruktion in Abhängigkeit der Verfügbarkeit von Programmcode und Daten auf verschiedenen Cache-Ebenen benötigt, sondern die Abfolge der Verarbeitungsschritte nichtdeterministische Elemente aufweist.

Diese lassen weder für einen externen Beobachter (es sei denn, dieser verfügte über ein exaktes physikalisches Modell sämtlicher Abläufe des zu evaluierenden Systems) noch für einen systemimmanenten Beobachter zu, dass die Abfolge von Ereignissen (genauer: durch diese induzierte Halbordnung) vorhergesagt werden kann. Gegeben eine Invariante J, welche durch eine Gruppe von Messungen bestimmt werden soll und ein Ereignis δ_{J_1} welches die zu erkennende Veränderung durch die Schadsoftware repräsentiert, sowie ein weiteres Ereignis δ_{J_2}, in dem die Veränderung durch die Schadsoftware maskiert bzw. rückgängig gemacht wird. Hieraus ergibt sich ein für die Erkennung kritischer Zeitraum $\Delta_J = [\delta_{J_1}, \delta_{J_2}]$

Sei weiterhin E die Menge der diskreten Messungen, welche durch den Erkennungsmechanismus über J und den Zeitraum der zu erkennenden Ereignisse durchgeführt wird. Für jede Teilmenge der Messungen $E^n \in E$ (d.h. jedes Ensemble von Messungen) besteht eine unabhängige Wahrscheinlichkeit $p(E^n)$ dass eine kritische Erkennungsperiode Δ_J mit E^n zusammenfällt.

Für eins gegebenes Messungs-Ensemble E^n ist, sofern alle Messungen hieraus zwischen den o.g. Ereignissen in Δ_J liegen, keine erfolgreiche Erkennung möglich (dies wird als *overlapped measurement set* bezeichnet). Selbiges ist ebenfalls der Fall, wenn alle Messungen außerhalb des Intervalles liegen (auch als *interspersed measurement set* bezeichnet). Sofern jedoch mindestens eine Messung E_i^n innerhalb des Intervalles sowie eine weitere Messung E_j^n außerhalb des Intervalles zu liegen kommt, liegt ein Erkennungsereignis D_J vor (dies wird als *combined measurement set* bezeichnet).

Die Wahrscheinlichkeit eines Erkennungsereignisses aufgrund einer gegebenen Teilmenge an Messungen in E kann durch die Darstellung des overlapped measurement set als p und des interspersed measurement set als q gefaßt werden. Für jede Teilmenge an Messungen E^n ist nun unter der o.g. Annahme, dass die Messungen unabhängig erfolgen und bei k Messungen in E^n p(x) als binomiale Wahrscheinlich-

keitsverteilung auf $P(E^n)$ aufzufassen und somit

$$P(D_J \mid E^n) = \sum_{i=1}^{k-1} \binom{k}{i} p^i q^{k-1}$$

Unter der Annahme, dass die Anzahl der nebenläufigen Messungen größer oder gleich der Anzahl der zur Verfügung stehenden Prozessorknoten ist, reduzieren sich dabei die Wahrscheinlichkeiten für overlapping bzw. interspersed measurement sets mit der Anzahl der Messungen und die Wahrscheinlichkeit eines combined measurement set steigt entsprechend. Bei geeigneter Wahl der Meßintervalle sowie der jeweils durchzuführenden Messungen lassen sich diese daher weitestgehend frei parametrisieren und ermöglichen eine zuverlässige Erkennung der unerwünschten Ereignisse.

Dies läßt jedoch bislang die Frage nach der geeigneten Auswahl der Meßpunkte bzw. der daraus resultierenden Invarianten und Relationen offen.

Zwar ist diese unter Voraussetzung, dass auch völlig unbekannte Angriffe erfaßt werden sollen a priori nicht beschränkt, jedoch ist zu beobachten, dass ungeachtet der konkreten Zielsetzung sowie der verwendeten Mechanismen i.d.R. Schadsoftware eine beschränkte Anzahl von Schnittstellen wie z.B. den Prozessorzustand, Elemente des Dateisystems, oder auch der Speicherverwaltung manipulieren müssen. Hieraus ergeben sich eine vergleichsweise kleine Menge an festzulegenden Relationen, welche anschließend durch Messungs-Ensembles überwacht werden müssen.

3 Nebenläufige Selbstberwachung

Neben der Möglichkeit, dass die Injektion von Schadcode hinreichend schnell erfolgt und somit die dabei wie im vorigen Abschnitt beschrieben notwendig auftretenden Inkonsistenzen nicht von konventionellen Angriffserkennungs-Mechanismen erfaßt werden, besteht bei Schadsoftware weiterhin noch das Risiko, dass Defensivmaßnahmen zu den primären Angriffszielen gehören. Zumeist wird ein Angreifer versuchen, die Protokollierungs- bzw. Angriffserkennungs-Mechanismen zu stören oder auch vollständig auszuschalten, um für die dann vorzunehmenden Manipulationen keine Erkennung fürchten zu müssen.

Der hier vorgestellte Mechanismus der nebenläufigen und nichtdeterministischen parallelen Sammlung von Messungs-Ensembles läßt sich jedoch unmittelbar auch zum Zweck der Selbstverteidigung einsetzen. Hierbei kann genutzt werden, dass die internen Zustände der Threads, welche Messungen durchführen und (siehe folgender Abschnitt) zusammenführen wohlbekannt sind und somit präzise Invarianten formuliert werden können.

Vermeidet man die Vorhaltung von Meßdaten an einzelnen kritischen Punkten (d.h. führt die Sammlung der Messung und ihre Auswertung parallel durch), kann man wiederum durch Erhöhung der Anzahl nebenläufiger Prozesse (indirekt somit auch der Prozessor-Kerne) die Wahrscheinlichkeit einer Kompromittierung wie im vorigen Abschnitt skizziert ebenfalls asymptotisch reduzieren. Hierbei ist zu beachten, dass mittels geeigneter kryptographischer Verfahren (Secure Multiparty Computations [18]) nach Abschluß einer Messung bzw. einer Erkennung diese Protokollereignisse fixiert werden können und auch eine spätere Kompromittierung des Systems dennoch diese erhalten bleiben läßt.

4 Zusammenführung von Messungen

Die zeitweilige Verletzung einer einzelnen Relation oder Invariante ist noch keine hinreichende Bedingung für die Erkennung eines Angriffs, da insbesondere auch das Betriebssystem selbst entsprechende Aktualisierungen vornimmt. Es ist daher erforderlich, mehrere Messungs-Ensembles bzw. die daraus resultierenden erkannten Verletzungen von Invarianten zusammenzuführen. Obgleich der Schwerpunkt der hier dargestellten Arbeit auf der Erfassung der Protokolldaten selbst und nicht in erster Linie auf die Erkennung von Angriffen abzielt, sei hier ein für die vorgestellten Messungen geeignetes effizientes Verfahren skizziert.

So eignet sich zur Repräsentation dynamischer Annahmen über kausale Relationen zwischen Ereignissen, welche durch die Hinzunahme weiterer Messungen widerlegt oder bestätigt werden insbesondere der Mechanismus des *Bayesian Belief Network* (BBN) [4,5]. Hierin werden in einem gerichteten azyklischen Graphen Variable als Knoten repräsentiert, deren Menge $V = \{V_0, \ldots, V_n\}$ einer gemeinsamen Wahrscheinlichkeitsverteilung unterliegen. Die Kanten im Graphen repräsentieren betigte Wahrscheinlichkeiten (z.B. $p(V_n = x \mid V_{n-1} = y \mid V_1 = z)$), welche aus kausalen Abhängigkeiten zwischen Knoten hergeleitet werden, wobei die Richtung des Graphen von Ursache (Hypothese) in Richtung Effekt (Beobachtung) zeigt. Dies erlaubt eine kompakte Darstellung, welche sich auch für weitere Analysen, etwa unter der Interpretation als Markov-Prozess, eignet.

Da die Modellierung als BBN lediglich eine Halbordnung auf den betrachteten Ereignissen erfordert, ist dies insbesondere für die hier betrachteten nebenläufigen Ereignisse von Vorteil; eine globale Ordnung auf den betrachteten Messungen ist ähnlich wie in verteilten Systemen nicht möglich.

5 Verwandte Arbeiten

Die Verwendung von Zufallsprozessen in der Erkennung von Angriffsereignissen wurde u.a. bereits in [2] und [14] untersucht; neben verschiedenen Software-Projekten ist die Untersuchung von Relationen innerhalb von Software u.a. in [13] dargestellt. Einige der Problemstellungen bei der Erkennung insbesondere von Rootkits werden in [1,3,6,8,9] behandelt; Owen et al. stellen in [11,12] Verfahren zur Erkennung von Angriffen bzw. Wiederherstellung des vormaligen Zustandes dar, wohingegen in [7,9] Angriffstypen beschrieben werden, welche derartige Verteidigungsmechanismen weitgehend umgehen können.

6 Fazit

Schadsoftware, welche unerkannt die Kontrolle über Arbeitsplatzrechner übernehmen und Daten nach Belieben manipulieren kann, stellt die Nicht-Abstreitbarkeit von Protokolldaten insgesamt und damit die gesamte Prozesskette revisionssicherer Protokollierung infrage.

Die Verhinderung derartiger Kompromittierungen oder zumindest die Erkennung eines Zeitpunktes, ab dem Protokolldaten nicht mehr vertrauenswürdig sind ist daher eine wesentliche Anforderung.

Unter Verwendung der zunehmend dominanten Mehrkern-Prozessorarchitektur können mittels gezielten Einsatzes von Nebenläufigkeit und nichtdeterministischer

Abläufe sich selbst überwachende Sensoren-Ensembles realisiert werden.

Entsprechende Experimente auf Grundlage eines Linux-Systems und unter Verwendung verschiedener Rootkits bestätigen die hier angestellten Überlegungen und ermöglichen eine zuverlässige Erkennung von Kompromittierungsversuchen [17].

Literatur

[1] CARRIER, B. Risks of Live Digital Forensic Analysis *Communications of the ACM 49 (2)*,. 56–61 (2006).

[2] CHUNG, S.P., MOK, A. On Random Inspection Based Intrusion Detection. *In Proceedings of the 8th International Symposium on Recent Advances in Intrusion Detection* (Seattle, WA, USA, Sep. 2005), Springer-Verlag, pp. 165–184.

[3] ANDERSON III, E. *A Demonstration of the Subversion Threat: Facing a Critical Responsibility in the Defense of Cyberspace*. M.Sc. Dissertation, U.S. Naval Postgraduate School (Monterey, CA, USA, 2002).

[4] PEARL, J. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, UK, 2000.

[5] PEARL, J. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufman Publishers, San Francisco, CA, USA, 1988.

[6] MYERS, P.A.. *Subversion: The Neglected Aspect of Computer Security*. M.Sc. Dissertation, U.S. Naval Postgraduate School (Monterey, CA, USA, 1980).

[7] HOGLUND, G. AND BUTLER, J. *Rootkits*. Addison-Wesley, Reading, MA, USA, 2005.

[8] RUTKOWSKA, J. Thoughts about Cross-view Based Rootkit Detection. Unveröffentlichtes Memorandum (2005).

[9] SZOR, P. *The Art of Computer Virus Research and Defense*. Addison-Wesley, Reading, MA, USA, 2005.

[10] HEASMAN, J. Implementing and Detecting an ACPI BIOS Root Kit. Vortrag anläßlich der Black Hat-Tagung (Las Vegas, NV, USA, 2005).

[11] GRIZZARD, J. AND OWEN, H.L.. On a µ-Kernel Based System Architecture Enabling Recovery from Rootkits. In *Proceedings of the First IEEE International Workshop on Critical Infrastructure Protection* (Darmstadt, Germany, Nov. 2005), IEEE Press, pp. 13–21.

[12] LEVINE, J., GRIZZARD, J. AND OWEN, H.L.. A Methodology to Detect and Characterize Kernel Level Rootkit Exploits Involving Redirection of the System Call Table. In *Proceedings of the Second IEEE International Workshop on Information Assurance* (Charlotte, NC, USA, Apr. 2004), IEEE Press, pp. 107–128.

[13] KRUEGEL, C., ROBERTSON, W., AND VIGNA, G.. Detecting Kernel-Level Rootkits Through Binary Audits. In *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC 2004)* (Tucson, AZ, USA, Dec. 2004), IEEE Press, pp. 91–100.

[14] UPPULURI, P. AND SEKAR, R. On Random Inspection Based Intrusion Detection. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection* (Davis, CA, USA, Oct. 2001), Springer-Verlag, pp. 172–189.

[15] KING, S.T., CHEN, P.M., WANG, Y.-M., VERBOWSKI, C., WANG, H.J. AND, LORCH, J.R. SubVirt: Implementing Malware with Virtual Machines. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy* (Oakland, CA, USA, May 2006), IEEE Press, pp. 314–327.

[16] WANG, Y.-M., BECK, D., VO, B., ROUSSEV, R. AND VERBOWSKI, C. Detecting Stealth Software with Strider Ghostbuster. In *Proceedings of the 2005 International Conference on Dependable Systems and Networks* (Oakland, CA, USA, May 2006), IEEE Press, pp. 368–377.

[17] MCEVOY T.R. AND WOLTHUSEN, S. Concurrent Non-Deterministic Monitoring of Operating System Kernel Properties. Technischer Bericht, Information Security Group, Department of Mathematics, Royal Holloway, University of London, UK (2007).

[18] LINDELL, Y. Composition of Secure Multi-Party Protocols: A Comprehensive Study. Springer-Verlag, Berlin, 2003.