Digital Business University
of Applied Sciences

Cyber- & IT-Security (M.Sc.)

Cloud Computing & Cloud Security

Wintersemester 2024/25

Hase und Igel

Aufgabe 4

Eingereicht von:
Elias Häussler

Matrikelnummer:
200094

Dozent:
Prof. Dr. Aymen Gatri

17. April 2025

Aufgabe

Im Paper *Logging to the Danger Zone: Race Condition Attacks and Defenses on System Audit Frameworks* (Paccagnella et al., 2020) geht es um *Race Conditions*. Erkläre genau, welche Annahme die Autoren dieses Papers außer Kraft setzen und inwieweit sie damit Race Conditions ausnutzen. Beschreibe den Ablauf des Tools *KennyLoggings* und wieso dieses Tool die Race Condition verhindert.

Diskutiere weiterhin, inwieweit der Dienst *Linux Audit* für Cloud-Systeme relevant ist und wie du zu deiner Einschätzung kommst.

Asynchrones Logging als Sicherheitsrisiko

Angreifer hinterlassen während eines Angriffs in der Regel Spuren, die zur Erkennung, Beseitigung sowie zur Identifikation der Angreifer beitragen können. Server-Logs beinhalten häufig eine Reihe solcher Spuren, da nahezu jedes Ereignis in einem entsprechenden Audit-Log festgeschrieben wird. Daher ist *Log-Tampering*, bei dem Angreifer versuchen, ihre Spuren noch während oder nach einem Angriff zu verwischen, längst zum Alltag geworden. Verschiedene Maßnahmen im Bereich des *Secure Logging* unterstützen dabei, Log-Tampering zu verhindern, indem Manipulationsresistenz und -erkennbarkeit gewährleistet werden sollen. Es gibt eine Vielzahl kommerzieller Anbieter, die sich auf diesen Bereich spezialisiert und fortschrittliche Lösungen implementiert haben.

Bietet *Secure Logging* ausreichend Schutz vor Log-Tampering?

Die Nutzung von Secure-Logging-Technologien gilt gemeinhin als eine sichere Maßnahme gegen Log-Tampering. Sie basiert auf der Annahme, dass das Auftreten eines Ereignisses gleichzeitig mit dessen Eintrag in das sichere Log erfolgt und ein potenzieller Angreifer keine Möglichkeit hat, einmal geschriebene Log-Einträge zu verändern oder gar zu entfernen.

Paccagnella et al. (2020) haben allerdings genau diese Annahme in ihrem Paper widerlegt. Zwar ist Secure Logging ein wirksames Mittel zum Schutz eines sicheren Audit-Logs vor Veränderung und Manipulation. Durch die asynchrone Natur von I/O[1]- und IPC[2]-Aktivitäten entsteht jedoch ein kurzes Zeitfenster, in dem ein Ereignis (z. B. ein Systemaufruf) zwar zugelassen, dessen Aufzeichnung aber noch nicht im Log protokolliert wurde. In dieser kurzen Zeitspanne ist es anfällig für Manipulationen, wobei die Zeit zwischen dem Auftreten eines Ereignisses und seiner Übergabe in das Log mit der Systemlast wächst und im Allgemeinen nicht zu vernachlässigen ist.

Race Conditions in der Zeit zwischen Ereignis und Logging

Die widerlegte Annahme der nicht vorhandenen bzw. zu vernachlässigenden Zeit zwischen dem Auftreten eines Ereignisses und dem zugehörigen Logging führt zu einer Reihe von Race Conditions. Der gesamte Prozess – vom Eintritt eines Ereignisses über das Erstellen der Log-Nachricht und deren Zwischenspeicherung im Puffer bis hin zur Übertragung in das (sichere) Log – ist angreifbar. Ein Angreifer kann Informationen über seinen Angriff aus den Nachrichtenpuffern entfernen, und zwar nachdem das Ereignis stattgefunden hat, aber noch bevor es ins Log geschrieben wurde. Er kann somit unbemerkt noch nicht verarbeitete Protokollereignisse abfangen und unterdrücken und so alle Spuren verwischen, die er ansonsten im Log hinterlassen würde. Entscheidend ist

[1] Input/Output
[2] Inter-process communication

hierbei, dass zwischen dem eigentlichen Ereignis und der Protokollierung eine Vielzahl unterschiedlicher Prozesse durchgeführt werden, teilweise auf Kernel-Ebene und damit im kritischen Bereich des Systems.

KennyLoggings als Log-Intercepting-Tool

Die skizzierte Grundproblematik liegt in der asynchronen Ausführung der Ereignisprotokollierung. Naheliegend ist daher der Wechsel auf eine rein synchrone Speicherung. Dies würde in der Praxis jedoch zu hohen Latenzen bei der Ausführung von Systemprozessen führen, was insbesondere bei modernen Systemen, die Millionen von Systemaufrufen pro Sekunde erzeugen, problematisch und auf Dauer nicht tragbar wäre.

Mit *KennyLoggings* haben die Autor*innen daher ein auf dem Konzept der *synchronen Integrität* aufbauendes, Kernel-basiertes Logging-System vorgestellt. Es soll einen manipulationssicheren Nachweis von Protokollereignissen garantieren, indem mithilfe kryptografischer Operationen ein Integritätsnachweis erzeugt und an das Protokollereignis angehängt wird. Dies erfolgt vollständig im Kernel, und zwar bevor die Nachricht in den Nachrichtenpuffer gereicht wird. Damit übernimmt das Tool die Kontrolle über den Nachrichtenpuffer im Kernel (*kernel log buffer*) und verhindert, dass kompromittierende Ereignisse vom Angreifer entfernt oder manipuliert werden, noch bevor sie vom Logging-Framework verarbeitet wurden:

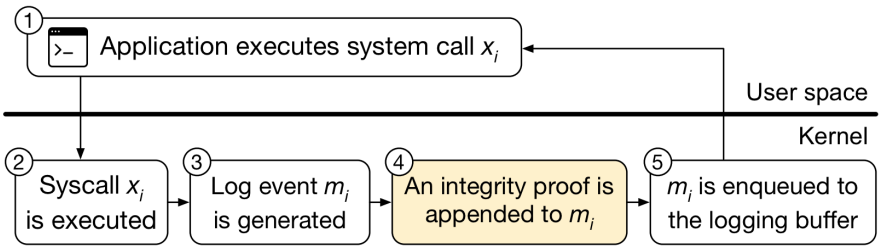

Abbildung 1: Workflow von *KennyLoggings*, aus: Paccagnella et al. (2020, S. 1558)

In Abbildung 1 ist dies schematisch auf Basis des *Linux Audit*-Dienstes dargestellt: Ein durch eine Anwendung durchgeführter Systemaufruf x_i (①) wird zur Verarbeitung an den Kernel übergeben, der diesen durch einen `audit_filter`-Hook leitet (②) und daraus eine Log-Nachricht m_i erstellt (③). Anschließend wird der kryptografische Integritätsnachweis erstellt und an die Nachricht angehängt, was durch Anpassungen an der `audit_log_end`-Funktion ermöglicht wird (④). Schließlich wird die Nachricht in den Nachrichtenpuffer übergeben, wo sie vom `kauditd`-Dienst verarbeitet werden kann (⑤). Im letzten Schritt übergibt der Kernel die Kontrolle wieder zurück an die Anwendung, um mit der weiteren Ausführung fortfahren zu können.

Indem das Tool direkt auf Kernel-Ebene ansetzt und noch vor Beendigung der ursprünglichen Systemausführung angewendet wird, können die vorgestellten Race Conditions somit verhindert werden.

Logging in Cloud-Systemen

Paccagnella et al. (2020) zeigen eindrucksvoll, wie wichtig ein funktionierendes Logging ist und welche Herausforderungen damit verbunden sind, um Log-Tampering möglichst zu vermeiden und gleichzeitig Angriffe erkennen und nachvollziehen zu können. Mit dem LAuS[3] – oder *Linux Audit* – existiert ein mächtiges System, das direkt im Linux-Kernel verankert ist. Es dient zur Überwachung und Protokollierung sicherheitsrelevanter Ereignisse wie der Änderung von Benutzerrechten, Dateizugriffen oder Systemaufrufen (siehe hierzu auch Abbildung 1, die einen Teil des Dienstes schematisch darstellt) (SUSE LINUX AG, 2004).

Relevanz von Logging in Cloud-Umgebungen

Auch – oder gerade – in Cloud-Systemen ist ein funktionierendes Logging von großer Bedeutung. Cloud-Infrastrukturen sind häufig Multimandantensysteme und daher besonders anfällig für schädliche und betrügerische Aktivitäten. Um ihren regulatorischen Pflichten nachzukommen, sind Cloud-Anbieter daher gut beraten, ein verlässliches Logging auf ihren Infrastrukturen zu etablieren.

Das LAuS ist hierbei prinzipiell eine sehr gute Wahl, denn es ist tief im System verankert und dadurch besonders effizient einsetzbar. Problematisch wird es jedoch im Zusammenhang mit Virtualisierungstechniken und in Public-Cloud-Umgebungen. Aufgrund ihres Aufbaus eignet sich Linux Audit beim Einsatz von Kubernetes oder Docker nur bedingt, da diese Techniken meist zu stark vom Kernel gekapselt sind, was eine effiziente Nutzung verhindert. Auch Cloud-Modelle wie PaaS[4] oder *Serverless* schließen die Nutzung von Linux Audit aus, da hierbei in der Regel kein Zugriff auf den Kernel besteht.

Grundsätzlich ist der Einsatz eines LAuS in Cloud-Umgebungen daher immer vom konkreten Einsatzzweck abhängig: Klassische Cloud-Infrastrukturen sind in aller Regel für den Einsatz eines LAuS geeignet, und dessen Nutzung ist in solchen Fällen auch sinnvoll. Moderne Virtualisierungslösungen hingegen eignen sich weniger gut, und auch andere Public-Cloud-Umgebungen sind aufgrund ihrer Einschränkungen zumeist eher ungeeignet.

Alternativen zu Linux Audit in Cloud-Infrastrukturen

Mit der Entwicklung moderner Cloud-Infrastrukturen haben die Anbieter ihren Fokus zunehmend auch auf den Betrieb sicherer Log-Mechanismen gelegt. Dienste wie *AWS*[5] *CloudTrail*, *Google Cloud Audit Logs* und *Azure Monitor* sind mächtige Monitoring-

[3] Linux Audit Subsystem
[4] Platform as a Service
[5] Amazon Web Services

Systeme, die detaillierte Einblicke in Systemprotokolle ermöglichen. Auch für Docker und Kubernetes existieren zahlreiche Konfigurations- und Integrationsmöglichkeiten interner und externer Dienste, mit denen ein umfangreiches Logging realisiert werden kann. Je nach Cloud-Architektur lässt sich ein LAuS auch mit einem solchen zentralen Log-Management kombinieren. Dies ermöglicht ein robustes, sicheres und gleichzeitig effizientes Logging der Cloud-Infrastruktur.

Literaturverzeichnis

Paccagnella, R., Liao, K., Tian, D., & Bates, A. (2020). Logging to the danger zone: Race condition attacks and defenses on system audit frameworks. *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 1551–1574.

SUSE LINUX AG. (2004). *Linux Audit-Subsystem Design Documentation for Kernel 2.6*. Version 0.1.

Eigenständigkeitserklärung

Ich trage die Verantwortung für die Qualität des Textes sowie die Auswahl aller Inhalte und habe sichergestellt, dass Informationen und Argumente mit geeigneten wissenschaftlichen Quellen belegt bzw. gestützt werden. Die aus fremden Quellen direkt oder indirekt übernommenen Texte, Gedankengänge, Konzepte, Grafiken usw. in meinen Ausführungen habe ich als solche eindeutig gekennzeichnet und mit vollständigen Verweisen auf die jeweilige Quelle versehen. Alle weiteren Inhalte dieser Arbeit (Textteile, Abbildungen, Tabellen etc.) ohne entsprechende Verweise stammen im urheberrechtlichen Sinn von mir.

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle sinngemäß und wörtlich übernommenen Textstellen aus fremden Quellen wurden kenntlich gemacht.

Die vorliegende Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Erklärung zu (gen)KI-Tools

Verwendung von (gen)KI-Tools

Ich versichere, dass ich mich (gen)KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Ich verantworte die Übernahme jeglicher von mir verwendeter Textpassagen vollumfänglich selbst. In der Übersicht verwendeter (gen)KI-Tools habe ich sämtliche eingesetzte (gen)KI-Tools, deren Einsatzform sowie die jeweils betroffenen Teile der Arbeit einzeln aufgeführt. Ich versichere, dass ich keine (gen)KI-Tools verwendet habe, deren Nutzung der Prüfer bzw. die Prüferin explizit schriftlich ausgeschlossen hat.

Hinweis: Sofern die zuständigen Prüfenden bis zum Zeitpunkt der Ausgabe der Aufgabenstellung konkrete (gen)KI-Tools ausdrücklich als nicht anzeige-/kennzeichnungspflichtig benannt haben, müssen diese nicht aufgeführt werden.

Ich erkläre weiterhin, dass ich mich aktiv über die Leistungsfähigkeit und Beschränkungen der unten genannten (gen)KI-Tools informiert habe und überprüft habe, dass die mithilfe der genannten (gen)KI-Tools generierten und von mir übernommenen Inhalte faktisch richtig sind.

Übersicht verwendeter (gen)KI-Tools

Die (gen)KI-Tools habe ich, wie im Folgenden dargestellt, eingesetzt.

(gen)KI-Tool	**Einsatzform**	**Betroffene Teile der Arbeit**
ChatGPT	Generierung von Zusammenfassungen verschiedener Literaturwerke	Gesamte Arbeit

Münster, 17. April 2025

Elias Häußler