

Christoph Scharnböck

UML - User Mode Linux

Ausbruch aus User Mode Linux

Bibliografische Information der Deutschen Nationalbibliothek:

Bibliografische Information der Deutschen Nationalbibliothek: Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.d-nb.de/> abrufbar.

Dieses Werk sowie alle darin enthaltenen einzelnen Beiträge und Abbildungen sind urheberrechtlich geschützt. Jede Verwertung, die nicht ausdrücklich vom Urheberrechtsschutz zugelassen ist, bedarf der vorherigen Zustimmung des Verlags. Das gilt insbesondere für Vervielfältigungen, Bearbeitungen, Übersetzungen, Mikroverfilmungen, Auswertungen durch Datenbanken und für die Einspeicherung und Verarbeitung in elektronische Systeme. Alle Rechte, auch die des auszugsweisen Nachdrucks, der fotomechanischen Wiedergabe (einschließlich Mikrokopie) sowie der Auswertung durch Datenbanken oder ähnliche Einrichtungen, vorbehalten.

Copyright © 2007 Diplomica Verlag GmbH
ISBN: 9783836629874

Christoph Scharnböck

UML - User Mode Linux

Ausbruch aus User Mode Linux

Christoph Scharnböck

UML - User Mode Linux

Ausbruch aus User Mode Linux

Christoph Scharnböck
UML - User Mode Linux
Ausbruch aus User Mode Linux

ISBN: 978-3-8366-2987-4

Herstellung: Diplomica® Verlag GmbH, Hamburg, 2009

Zugl. Fachhochschule Hagenberg, Hagenberg, Österreich, Bachelorarbeit, 2007

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtes.

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Die Informationen in diesem Werk wurden mit Sorgfalt erarbeitet. Dennoch können Fehler nicht vollständig ausgeschlossen werden und der Verlag, die Autoren oder Übersetzer übernehmen keine juristische Verantwortung oder irgendeine Haftung für evtl. verbliebene fehlerhafte Angaben und deren Folgen.

© Diplomica Verlag GmbH
<http://www.diplomica.de>, Hamburg 2009

Inhaltsverzeichnis

Motivation	vii
Kurzfassung	x
Abstract	xi
1 Grundlagen	1
1.1 Linux-Kernel	1
1.1.1 Einleitung	1
1.1.2 Aufbau und Funktionsweise des monolithischen Kernels	3
1.2 Prozessmanagement	9
1.2.1 Prozesse vs. Threads	9
1.2.2 Erstellen & Beenden von Prozessen	11
1.2.3 Prozesszustände	13
1.2.4 User- & Kernelmode	13
1.2.5 Interrupts	14
1.3 Speichermanagement	16
1.3.1 Speicheradressierung	16
1.3.2 Paging	18
1.4 I/O	20
1.4.1 Hardwareschichten	20
1.4.2 Softwareschichten	23
1.5 Virtual Machines	26
1.5.1 Techniken	26
1.5.2 Virtual Machine Monitor	28
1.5.3 CPUs mit Virtualisierungstechnologien	28
2 UML Theorie	30
2.1 Architektur	30
2.1.1 Aufbau	30
2.1.2 Systemcalls	31
2.1.3 Traps	33

2.2	Managmentkonsole	34
2.2.1	Aufbau	34
2.3	Ausführungsmodi	35
2.3.1	tt-Modus	36
2.3.2	Skas3-Modus	38
2.3.3	Skas0-Modus	41
2.4	Speichermanagement	42
2.4.1	Hauptspeicher	42
2.4.2	Filesysteme	43
2.4.3	hostfs	45
2.4.4	humfs	46
2.5	Security	48
2.5.1	Ausbruchsmöglichkeiten	48
2.5.2	Chroot	49
2.5.3	Ausbruch aus einem <i>UML</i> -Jail	49
3	UML-Praxis - Ausbruchsmöglichkeiten	51
3.1	Ausbruch durch das Filesystem	51
3.1.1	Vorgehensweise	52
3.1.2	Abwehr	53
3.2	Kernelmodul	53
3.2.1	Idee	54
3.2.2	Charakteristik von Modulen	54
3.2.3	Funktion	55
3.2.4	Modulerstellung	56
3.3	UML-Hauptspeicher	58
3.3.1	Idee	58
3.3.2	Ablauf	58
3.4	Systemcallhacking	59
3.4.1	lcall	60
3.4.2	ptrace	60
3.5	Shellcodeinjection	63
3.5.1	Vorgehensweise	63
3.5.2	Aufbau	64
3.5.3	Einschleusung	64
4	Fazit und Ausblick	66
5	Glossar	69
A	Zusatzinformation	72
A.1	UML-Theorie	72
A.1.1	Sysrq-Kommandos	72
A.1.2	Terminal I/O	72

<i>INHALTSVERZEICHNIS</i>	vi
---------------------------	----

A.1.3 tmpfs-Performancetest	73
---------------------------------------	----

Literaturverzeichnis	80
-----------------------------	-----------

Motivation

User-Mode-Linux, abgekürzt als *UML*¹, wurde von Jeff Dike im Jahre 1999 entwickelt. Es handelt sich dabei um ein Produkt aus der Virtualisierung. Der Wunsch, Betriebssysteme unabhängig innerhalb anderer Betriebssysteme zu betreiben, wurde mit *UML* auf einem bisher noch nicht existierenden Lösungsweg realisiert. Bisher musste, unabhängig von der eingesetzten Technologie, eine Zwischenschicht die Trennung der Betriebssysteme vornehmen. Die *UML*-Architektur ist so konstruiert, dass auf einen performance-lastigen Layer verzichtet werden kann. Damit dieses Konzept auch auf existierende Betriebssysteme umgesetzt werden kann, ist hier schon eine erste Einschränkung zu sehen. *UML* unterstützt lediglich Linux-Betriebssysteme.

UML kann von zwei unterschiedlichen Standpunkten analysiert und beurteilt werden. Die häufig vorkommende Feststellung „UML ist ein Betriebssystem, das wie ein Programm auf einem Betriebssystem läuft“ liefert schon eine grundlegende Definition von dem, was *UML* darstellt. Aus dem Betrachtungswinkel eines Benutzers am Hostsystem ist *UML* ein Benutzerprozess, auf den ersten Blick nicht vom Prozess einer anderen Anwendung zu unterscheiden. Aus anderem Betrachtungswinkel, dem des Benutzers innerhalb *UML*, ist *UML* ein vollwertiges Betriebssystem, das zunächst nicht von einem Hostbetriebssystem unterscheidbar ist.

Der Übergang vom Betriebssystem zum Benutzerprogramm muss, da kein Virtualisierungslayer existiert, vom *UML*-Kernel übernommen werden. Um einen Linuxkernel zu einem *UML*-Linuxkernel umzufunktionieren, muss bei dessen Übersetzung lediglich die eigens dafür entwickelte *User-Mode-Architektur* anstelle z.B. der *x86*-Architektur verwendet werden. Zugriffe, die ein gewöhnlicher Kernel an der Hardware durchführt, werden bei einem *UML*-Kernel durch die *UML-Arch*-Schnittstelle simuliert. Diese Schicht leitet die Anfragen an den Hostkernel als gewöhnlicher Usermode-Prozess weiter.

Gerade durch diese transparente Architektur bietet *UML* interessante Einsatzgebiete.

¹Nicht zu verwechseln mit der Abkürzung für „Unified Modeling Language“, einer standardisierten Sprache zur Modellierung von Systemen.