

VU Research Portal

Building a dependable operating system: fault tolerance in MINIX 3

Herder, J.N.

2010

document version

Publisher's PDF, also known as Version of record

[Link to publication in VU Research Portal](#)

citation for published version (APA)

Herder, J. N. (2010). *Building a dependable operating system: fault tolerance in MINIX 3*. [PhD-Thesis - Research and graduation internal, Vrije Universiteit Amsterdam].

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

E-mail address:

vuresearchportal.ub@vu.nl

Contents

ACKNOWLEDGEMENTS	xiii
SAMENVATTING	xv
1 GENERAL INTRODUCTION	1
1.1 The Need for Dependability	2
1.2 The Problem with Device Drivers	4
1.3 Why do Systems Crash?	6
1.3.1 Software Complexity	6
1.3.2 Design Flaws	8
1.4 Improving OS Dependability	9
1.4.1 A Modular OS Design	10
1.4.2 Fault-tolerance Strategies	12
1.4.3 Other Benefits of Modularity	13
1.5 Preview of Related Work	14
1.6 Focus of this Thesis	16
1.7 Outline of this Thesis	18
2 ARCHITECTURAL OVERVIEW	19
2.1 The MINIX Operating System	19
2.1.1 Historical Perspective	19
2.1.2 Multiserver OS Structure	21
2.1.3 Interprocess Communication	22
2.2 Driver Management	24
2.3 Isolating Faulty Drivers	26
2.3.1 Isolation Architecture	26
2.3.2 Hardware Considerations	30
2.4 Recovering Failed Drivers	32
2.4.1 Defect Detection and Repair	32
2.4.2 Assumptions and Limitations	34
2.5 Fault and Failure Model	35

3	FAULT ISOLATION	37
3.1	Isolation Principles	37
3.1.1	The Principle of Least Authority	37
3.1.2	Classification of Privileged Operations	38
3.1.3	General Rules for Isolation	41
3.2	User-level Driver Framework	42
3.2.1	Moving Drivers to User Level	42
3.2.2	Supporting User-level Drivers	43
3.3	Isolation Techniques	44
3.3.1	Restricting CPU Usage	44
3.3.2	Restricting Memory Access	45
3.3.3	Restricting Device I/O	50
3.3.4	Restricting IPC	51
3.4	Case Study: Living in Isolation	54
4	FAILURE RESILIENCE	57
4.1	Defect Detection Techniques	57
4.1.1	Unexpected Process Exits	58
4.1.2	Periodic Status Monitoring	58
4.1.3	Explicit Update Requests	59
4.2	On-the-fly Repair	60
4.2.1	Recovery Scripts	60
4.2.2	Restarting Failed Components	61
4.2.3	State Management	63
4.3	Effectiveness of Recovery	65
4.3.1	Recovering Device Drivers	66
4.3.2	Recovering System Servers	70
4.4	Case Study: Monitoring Driver Correctness	70
4.5	Case Study: Automating Server Recovery	73
5	EXPERIMENTAL EVALUATION	75
5.1	Software-implemented Fault Injection	75
5.1.1	SWIFI Test Methodology	75
5.1.2	Network-device Driver Results	79
5.1.3	Block-device Driver Results	85
5.1.4	Character-device Driver Results	87
5.2	Performance Measurements	89
5.2.1	Costs of Fault Isolation	89
5.2.2	Costs of Failure Resilience	94
5.3	Source-code Analysis	96
5.3.1	Evolution of MINIX 3	96
5.3.2	Evolution of Linux 2.6	99

6	RELATED WORK	101
6.1	In-kernel Sandboxing	101
6.1.1	Hardware-enforced Protection	102
6.1.2	Software-based Isolation	104
6.2	Virtualization Techniques	107
6.2.1	Full Virtualization	107
6.2.2	Paravirtualization	109
6.3	Formal Methods	111
6.3.1	Language-based Protection	112
6.3.2	Driver Synthesis	115
6.4	User-level Frameworks	117
6.4.1	Process Encapsulation	117
6.4.2	Split-driver Architectures	120
6.5	Comparison	123
7	SUMMARY AND CONCLUSION	125
7.1	Summary of this Thesis	125
7.1.1	Problem Statement and Approach	125
7.1.2	Fault-tolerance Techniques	128
7.2	Lessons Learned	130
7.2.1	Dependability Challenges	131
7.2.2	Performance Perspective	132
7.2.3	Engineering Effort	133
7.3	Epilogue	135
7.3.1	Contribution of this Thesis	135
7.3.2	Application of this Research	136
7.3.3	Directions for Future Research	137
7.4	Availability of MINIX 3	139
	REFERENCES	141
	ABBREVIATIONS	161
	PUBLICATIONS	163
	BIOGRAPHY	165