

Diss. ETH No. 20442

Safe Loading and Efficient Runtime Confinement: A Foundation for Secure Execution

A dissertation submitted to
ETH ZURICH

for the degree of
Doctor of Sciences

presented by
Mathias J. Payer
Master of Science ETH in Computer Science, ETH Zurich
born April 29, 1981
citizen of the Principality of Liechtenstein

accepted on the recommendation of
Prof. Dr. Thomas R. Gross, examiner
Prof. Dr. Srdjan Capkun, co-examiner
Prof. Dr. Steve Hand, co-examiner

2012

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | Attack model | 3 |
| 1.2 | Requirements for a secure execution platform | 3 |
| 1.3 | The foundation of a secure dynamic execution platform | 4 |
| 1.4 | Thesis statement | 7 |
| 1.5 | Libdetox | 7 |
| 1.6 | Contributions | 8 |
| 1.7 | Publications | 8 |
| 1.8 | Outline | 9 |
| 2 | Background information | 11 |
| 2.1 | Attack vectors | 11 |
| 2.2 | Exploit classes | 12 |
| 2.3 | Comparison to Erlingssorrs attack classification | 20 |
| 2.4 | Security of the standard loader | 21 |
| 2.5 | Binary translation | 23 |
| 2.6 | Summary | 24 |
| 3 | Related work | 25 |
| 3.1 | Binary translation | 25 |
| 3.2 | Software-based Fault Isolation (SFI) | 28 |
| 3.3 | System call authorization | 29 |
| 3.4 | Full-system virtualization | 30 |
| 3.5 | Static program verification | 30 |
| 3.6 | Secure compiler extensions | 33 |
| 3.7 | Summary of different protection techniques | 33 |

| | | |
|----------|---|------------|
| 4 | Design and security guidelines | 35 |
| 4.1 | Safe loading in a trusted runtime environment | 37 |
| 4.2 | Software-based fault isolation layer. | 40 |
| 4.3 | Dynamic Control Flow Integrity (CFI). | 43 |
| 4.4 | Model generation for dynamic CFI. | 44 |
| 4.5 | Policy-based system call interposition. | 48 |
| 4.6 | Summary. | 51 |
| 5 | System architecture and implementation | 53 |
| 5.1 | Secure loader. | 54 |
| 5.2 | A generic dynamic binary translator. | 60 |
| 5.3 | Software-based fault isolation layer. | 66 |
| 5.4 | Dynamic control flow integrity. | 70 |
| 5.5 | Policy-based system call authorization. | 72 |
| 5.6 | Discussion. | 74 |
| 6 | Evaluation | 75 |
| 6.1 | Security evaluation. | 75 |
| 6.2 | SPEC CPU 2006 characteristics. | 81 |
| 6.3 | Libdetox performance evaluation. | 82 |
| 6.4 | Control flow integrity using ELF information. | 85 |
| 6.5 | System call policies. | 88 |
| 6.6 | Apache case study. | 91 |
| 7 | Case study: dynamic race detection | 93 |
| 7.1 | Attack model and background information. | 95 |
| 7.2 | The DynaRace approach. | 96 |
| 7.3 | Implementation. | 101 |
| 7.4 | Implementation alternatives. | 106 |
| 7.5 | Evaluation. | 108 |
| 7.6 | Related work to file-based race detection. | 116 |
| 7.7 | Limitations and weaknesses. | 117 |
| 7.8 | Summary. | 118 |
| 8 | Future directions | 119 |
| 8.1 | Compiler-based CFG generation. | 119 |

| | | |
|----------|---|------------|
| 8.2 | A compiler-driven approach to system call policies. | 120 |
| 8.3 | I/O purification extension. | 120 |
| 8.4 | Dynamic patching. | 121 |
| 9 | Concluding remarks | 123 |
| 9.1 | Summary and contributions. | 123 |
| 9.2 | Expandability. | 124 |
| A | x86 ISA | 125 |
| B | ELF format and the Linux loader | 126 |
| C | System call interface | 129 |
| C.1 | Argument passing. | 129 |
| C.2 | Software interrupts. | 130 |
| C.3 | sysenter instruction. | 130 |
| D | CFI micro benchmarks | 131 |
| E | Libdetox evaluation with additional optimizations | 132 |
| | References | 134 |
| | Acronyms | 141 |
| | Curriculum Vitae | 142 |