VMI-PL: A monitoring language for virtual platforms using virtual machine introspection

Florian Westphal [1,2] Stefan Axelsson [1] Christian Neuhaus [2]
Andreas Polze [2]

[1]Blekinge Institute of Technology
Karlskrona, Sweden

[2]Hasso Plattner Institute
Potsdam, Germany

August 5, 2014

Outline

Virtual Machine Introspection (VMI)

"Virtual machine introspection (VMI) describes the method of monitoring and analyzing the state of a virtual machine from the hypervisor level."

— Jonas Pfoh [Pfoh et al., 2009]

Virtual Machine Introspection (VMI)

> "Virtual machine introspection (VMI) describes the method of monitoring and analyzing the state of a virtual machine from the hypervisor level."
>
> — Jonas Pfoh [Pfoh et al., 2009]

- First Described by Garfinkel and Rosenblum [Garfinkel et al., 2003]

Virtual Machine Introspection (VMI)

> "Virtual machine introspection (VMI) describes the method of monitoring and analyzing the state of a virtual machine from the hypervisor level."
>
> — Jonas Pfoh [Pfoh et al., 2009]

- First Described by Garfinkel and Rosenblum [Garfinkel et al., 2003]
- Semantic gap problem [Chen et al., 2001]

VMI Techniques

VMI Techniques

Data-based techniques:

VMI Techniques

Data-based techniques:

- Memory introspection

VMI Techniques

Data-based techniques:

- Memory introspection
- Register introspection

VMI Techniques

Data-based techniques:

- Memory introspection
- Register introspection

Event-based techniques:

VMI Techniques

Data-based techniques:

- Memory introspection
- Register introspection

Event-based techniques:

- Monitor writes to certain registers

VMI Techniques

Data-based techniques:

- Memory introspection
- Register introspection

Event-based techniques:

- Monitor writes to certain registers
- Monitor system and user defined interrupts

VMI Techniques

Data-based techniques:

- Memory introspection
- Register introspection

Event-based techniques:

- Monitor writes to certain registers
- Monitor system and user defined interrupts

Stream-based techniques:

VMI Techniques

Data-based techniques:

- Memory introspection
- Register introspection

Event-based techniques:

- Monitor writes to certain registers
- Monitor system and user defined interrupts

Stream-based techniques:

- Inspect network traffic

VMI Techniques

Data-based techniques:

- Memory introspection
- Register introspection

Event-based techniques:

- Monitor writes to certain registers
- Monitor system and user defined interrupts

Stream-based techniques:

- Inspect network traffic
- Monitor keyboard input

Outline

VMI-PL - Overview I

VMI-PL:

VMI-PL - Overview I

VMI-PL:

- Data probes
- Event probes
- Stream probes

VMI-PL - Overview I

VMI-PL:

- Data probes
- Event probes
- Stream probes
- Filters
- Reconfiguration instructions

VMI-PL - Overview I

VMI-PL:

- Data probes
- Event probes
- Stream probes
- Filters
- Reconfiguration instructions
- Configuration

VMI-PL - Overview II

```
Configuration{
    <configuration-item>
}
<event-probe>(<parameters>){
    <data-probe>(<parameters>)
    <filter>(<parameters>){
        <data-probe>(<parameters>)
    }
}
<stream-probe>(<parameters>)
```

VMI-PL - Language Constructs I

VMI-PL - Language Constructs I

Data Probes:

- `ReadMemory(<virtual address> <size>, [<file-path> | <stream-id>])`

VMI-PL - Language Constructs I

Data Probes:

- `ReadMemory(<virtual address> <size>, [<file-path> | <stream-id>])`
- `WriteToMemoryAt(<register>, <offset>, <value>)`

VMI-PL - Language Constructs I

Data Probes:

- `ReadMemory(<virtual address> <size>, [<file-path> | <stream-id>])`
- `WriteToMemoryAt(<register>, <offset>, <value>)`
- `ProcessList(<field>, [<file-path> | <stream-id>])`

VMI-PL - Language Constructs I

Data Probes:

- `ReadMemory(<virtual address> <size>, [<file-path> | <stream-id>])`
- `WriteToMemoryAt(<register>, <offset>, <value>)`
- `ProcessList(<field>, [<file-path> | <stream-id>])`

Event Probes:

- `CRWrite(<number>)`

VMI-PL - Language Constructs I

Data Probes:

- `ReadMemory(<virtual address> <size>, [<file-path> | <stream-id>])`
- `WriteToMemoryAt(<register>, <offset>, <value>)`
- `ProcessList(<field>, [<file-path> | <stream-id>])`

Event Probes:

- `CRWrite(<number>)`
- `Syscall([<syscall number> | -])`

VMI-PL - Language Constructs I

Data Probes:

- ReadMemory(<virtual address> <size>, [<file-path> | <stream-id>])
- WriteToMemoryAt(<register>, <offset>, <value>)
- ProcessList(<field>, [<file-path> | <stream-id>])

Event Probes:

- CRWrite(<number>)
- Syscall([<syscall number> | -])

Stream Probes:

- CaptureNetwork(<mac address>, [<file-path> | <stream-id>])

VMI-PL - Language Constructs I

Data Probes:

- ReadMemory(<virtual address> <size>, [<file-path> | <stream-id>])
- WriteToMemoryAt(<register>, <offset>, <value>)
- ProcessList(<field>, [<file-path> | <stream-id>])

Event Probes:

- CRWrite(<number>)
- Syscall([<syscall number> | -])

Stream Probes:

- CaptureNetwork(<mac address>, [<file-path> | <stream-id>])
- CaptureKeyboardInput([<file-path> | <stream-id>])

VMI-PL - Language Constructs II

VMI-PL - Language Constructs II

Filters:

- `RegisterHasValue(<register>, <value>)`

VMI-PL - Language Constructs II

Filters:

- `RegisterHasValue(<register>, <value>)`

Reconfiguration Instructions:

- `Pause`

VMI-PL - Language Constructs II

Filters:

- `RegisterHasValue(<register>, <value>)`

Reconfiguration Instructions:

- `Pause`
- `Reconfigure(<event probe id>)`

VMI-PL - Demo I

```
Configuration{
    ProcessListHead: 0xc17cdfe0
    TasksOffset: 440
    PIDOffset: 520
    ProcessNameOffset: 744
    MMStructOffset: 468
    ExeFileOffset: 444
    DEntryOffset: 12
    ParentOffset: 16
    DNameOffset: 28
    PGDOffset: 40
}
CRWrite(3){
    ReadRegister(CR3, #demo)
}
ExecuteAt(0xc104f060){
    ProcessList(PID, NAME, PATH, PGDP, #demo)
    ReadRegister(CR3, #demo)
}
CaptureNetwork(00:16:35:AF:94:4B, #demo)
```

VMI-PL - Demo II

Demo

Outline

Evaluation - Measurement Scenarios

- Without monitoring support

Evaluation - Measurement Scenarios

- Without monitoring support
- Monitoring support enabled (*VMI-PL*, *VProbes*)

Evaluation - Measurement Scenarios

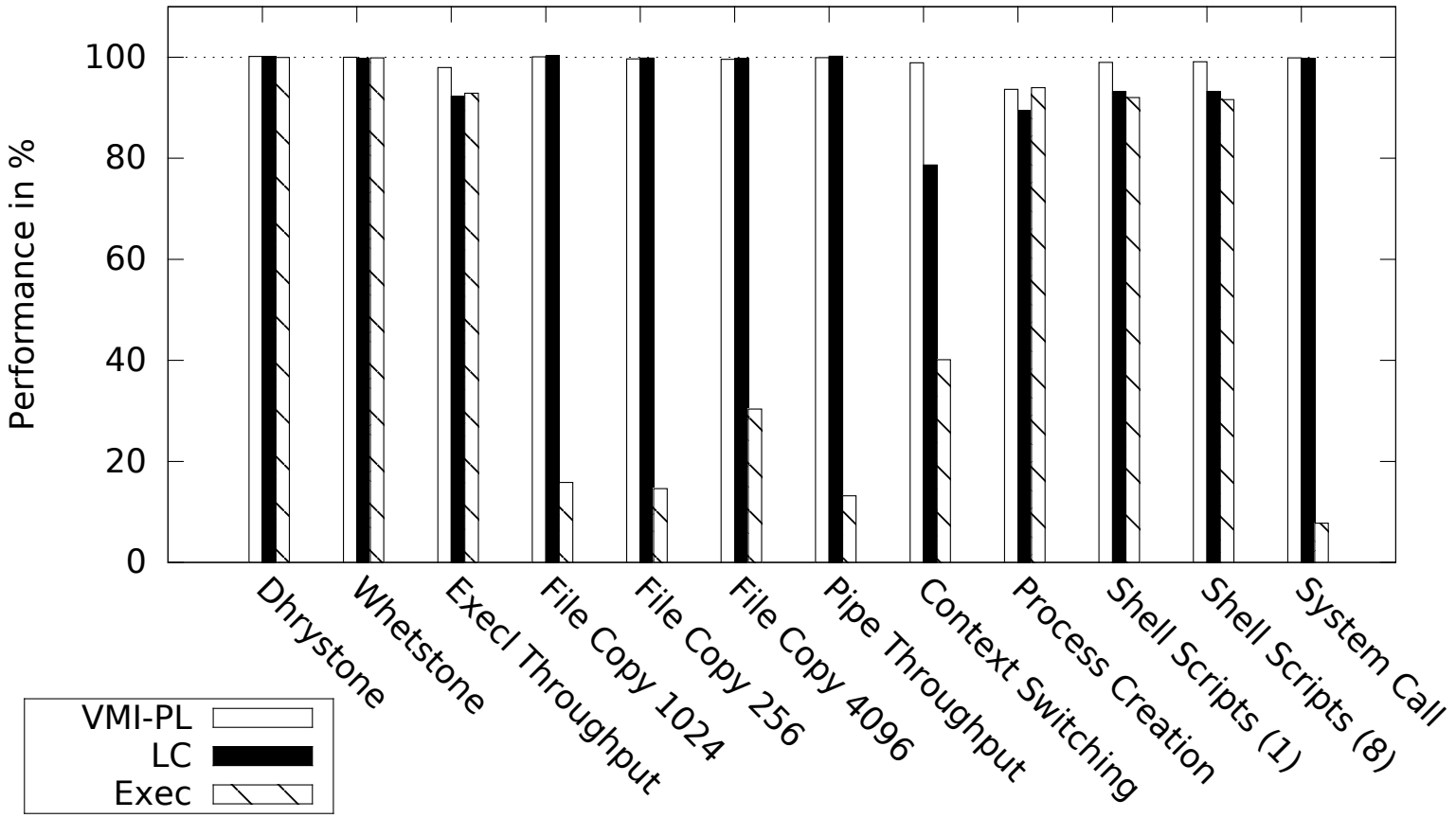
- Without monitoring support
- Monitoring support enabled (*VMI-PL*, *VProbes*)
- Process life cycle monitoring (*LC*)

Evaluation - Measurement Scenarios

- Without monitoring support
- Monitoring support enabled (*VMI-PL*, *VProbes*)
- Process life cycle monitoring (*LC*)
- Process execution monitoring (*Exec*)

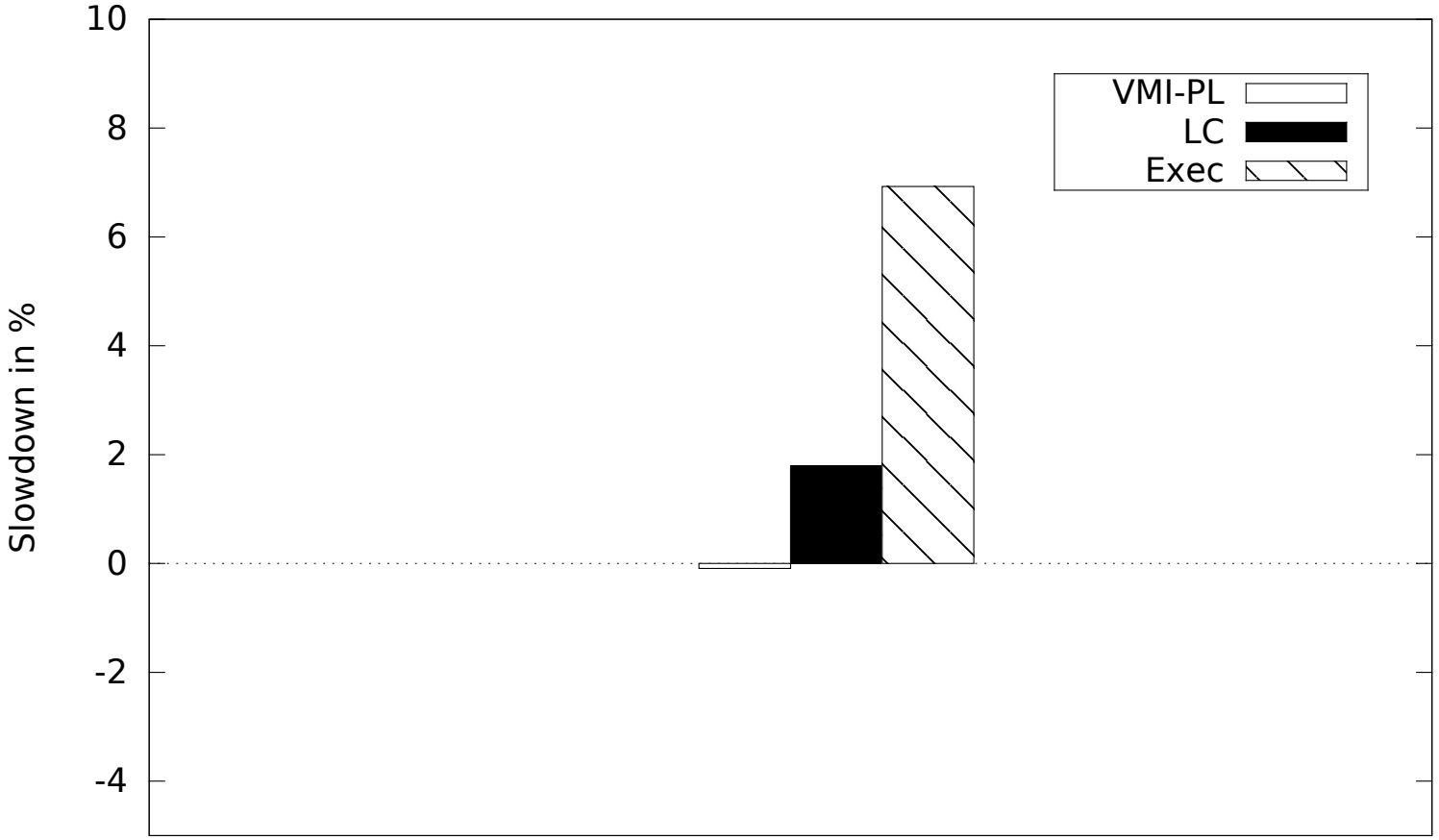
Evaluation - VMI-PL Performance I

UnixBench Benchmark



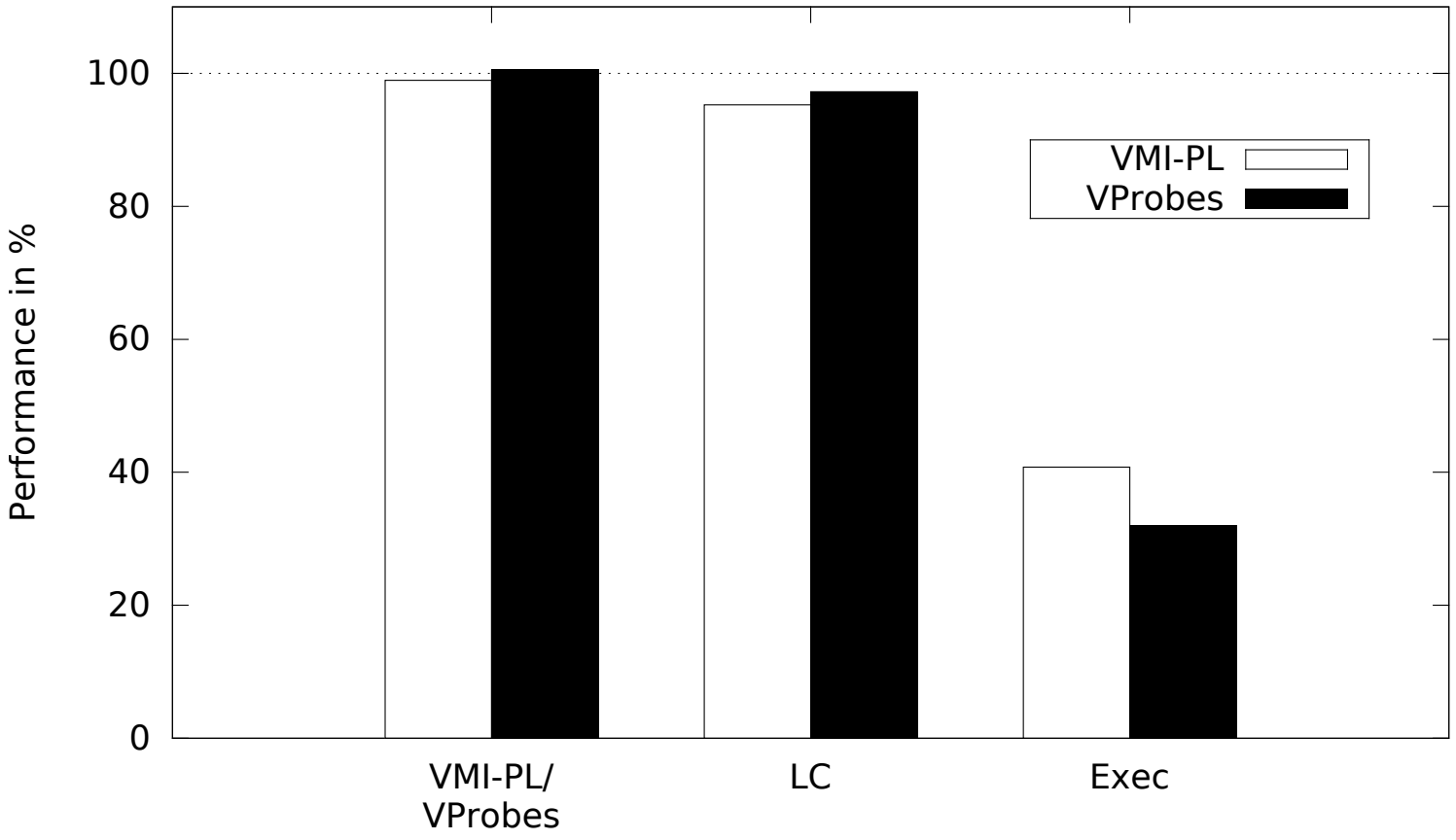
Evaluation - VMI-PL Performance II

VMI-PL Apache Build Slowdown



Evaluation - Comparison

UnixBench Benchmark



Outline

Conclusion

- Classification scheme for VMI techniques

Conclusion

- Classification scheme for VMI techniques
- Simple description language for VMI

Conclusion

- Classification scheme for VMI techniques
- Simple description language for VMI
  - Support for data, event, and stream-based techniques

Conclusion

- Classification scheme for VMI techniques
- Simple description language for VMI
  - Support for data, event, and stream-based techniques
  - Support for VM state manipulation

Conclusion

- Classification scheme for VMI techniques
- Simple description language for VMI
  - Support for data, event, and stream-based techniques
  - Support for VM state manipulation
  - Support for retrieving operating system level information

Conclusion

- Classification scheme for VMI techniques
- Simple description language for VMI
  - Support for data, event, and stream-based techniques
  - Support for VM state manipulation
  - Support for retrieving operating system level information
- Prototype for KVM
  - Available at `https://github.com/FlorianWestphal/VMI-PL`

References

J. Pfoh, Ch. Schneider, C. Eckert (2009)
A Formal Model for Virtual Machine Introspection.
In Proc. of the 1st ACM workshop on Virtual machine security 1 – 10.

T. Garfinkel, M. Rosenblum (2003)
A Virtual Machine Introspection Based Architecture for Intrusion Detection.
In Proc. Network and Distributed Systems Security Symposium 191 – 206.

P.M. Chen, B.D. Noble (2001)
When virtual is better than real [operating system relocation to virtual machines].
Hot Topics in Operating Systems, 2001. Proceedings of the Eighth Workshop on 133–138.