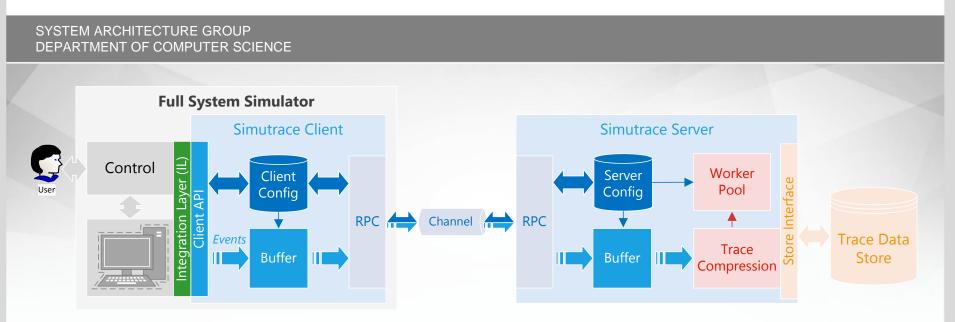


## **Efficient Full System Memory Tracing with Simutrace**

### GI Fachgruppentreffen Betriebssysteme (BS) 2014

Thorsten Gröninger, Marc Rittinghaus, Frank Bellosa

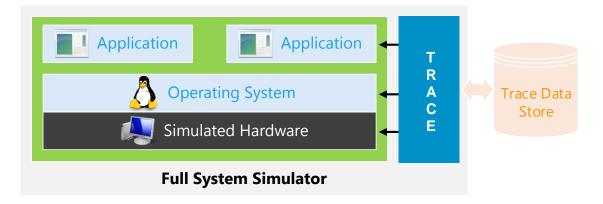


KIT – University of the State of Baden-Wuerttemberg and National Research Center of the Helmholtz Association

## **Motivation**



- Operating system performance analysis
  - Application and kernel interaction
  - Memory access patterns
  - Cache efficiency



### Want to trace and do offline analysis

- Full system simulation is slow (100x)
- Repeatability/reproducibility of experiments/results

## **Memory Tracing**



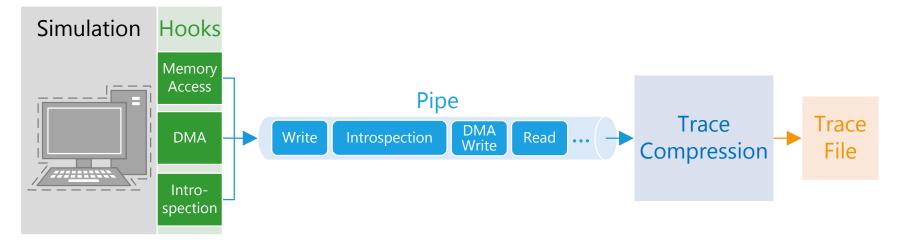
- Want: Record memory accesses and introspection data
  - Correlate operations with processes, memory areas, etc.
- Challenges
  - High event rate: approx. 150 MiB/s (QEmu single-threaded)
  - High amount of data:

Kernel Build	SPEC.h264 (w)	SPEC.gcc
3.6 TiB	11 TiB	8 TiB
150 bil. Entries	360 bil. Entries	310 bil. Entries

### We need to compress traces



## **Memory Tracing**



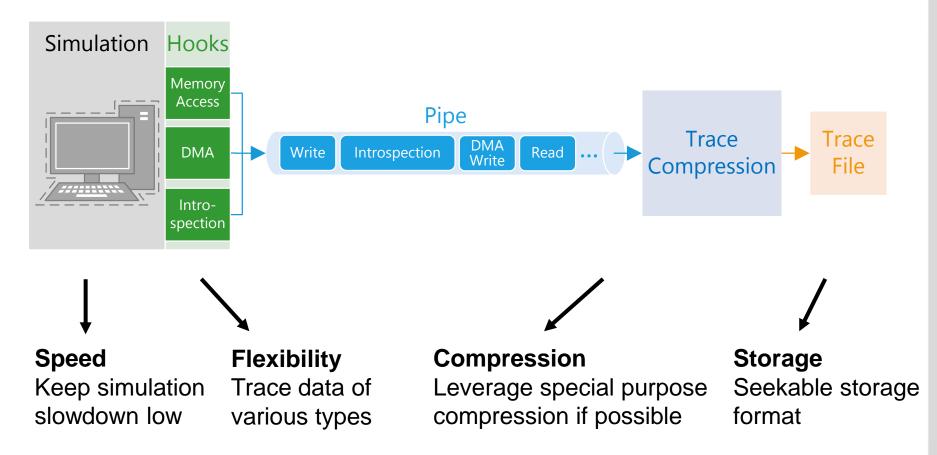
- Simple tracer: pipe data into compressor
  - ± Kernel build (3.6 TiB): 700 GiB (bzip2), 800 GiB (deflate)
  - Simulation slowdown: 7x (deflate)

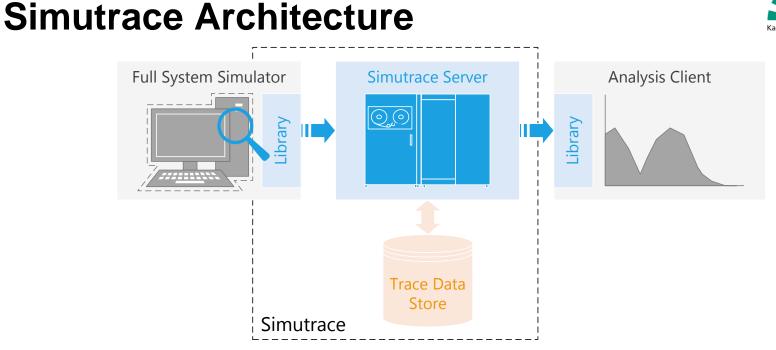
Simulation	Trace Compression and Storage	
10 %	90 %	

### We need a more sophisticated approach

## **Simutrace Design Goals**







- Client-server architecture
  - Clients submit or query data
  - Server processes traces and manages storage
- Library in client manages connection
  - Trace data exchange over shared memory or sockets



## Simutrace – Flexibility

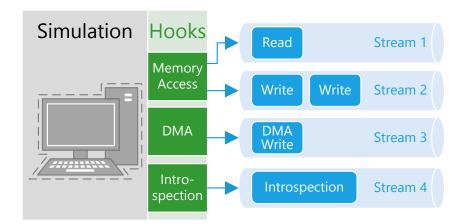
Want: Trace data of various types

- Challenges
  - Entries of different size and compressibility are interleaved
  - Varying number of entries per type

### Separate entries of different type

### Streams

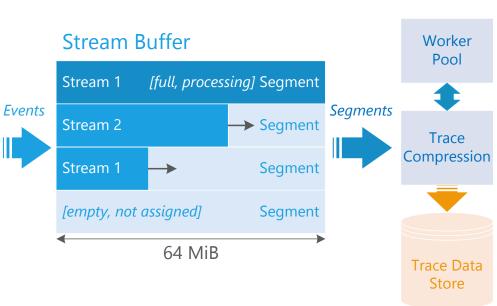
- Contain only one type
- Group semantically connected entries
- Ease type specific compression
- Ease addition of further data



## Simutrace – Speed

Want: Keep simulation slowdown low

- Asynchronous compression
  - Shared buffer (zero-copy)
  - Segment granularity
- Parallel compression
  - Scales with submission rate



- Simulation runs at near full speed
  - Simulation slowdown: 1.1x

# SimulationTracing90 %10 %

Karlsruhe Institute of



## Simutrace – Compression



Want: Leverage special purpose compression schemes

- Compression scheme depends on stream's data type
  - General purpose as default (LZMA)
- Memory access: Modified version of VPC4 (SVPC)
  - Leverages memory access locality
  - Combines prediction-based compression with LZMA
  - Improves compression ratio and speed
- Detailed traces become manageable

	Kernel Build	SPEC.gcc	memtest
RAW	3.6 TiB	8 TiB	308 GiB
Simutrace	110 GiB (1:32)	70 GiB (1:114)	98 MiB (1:3142)

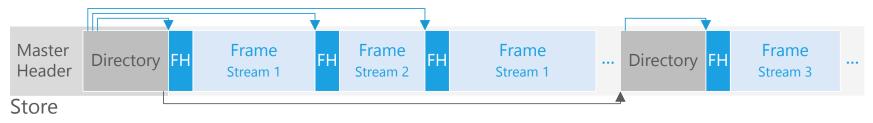


## Simutrace – Storage

Want: Seekable storage format

- Record metadata in trace
  - Simulation time, wall clock time, etc.
  - Index to find position in file

### We need partial decompression



- Trace: List of compressed segments
  - + Partial decompression
  - Segments have varying size
  - Segments may be in wrong order

Directory for fast discovery

#### FrameBuffer Content:

[	0.274781]	NetLabel: domain hash size = 128
[ [ [	0.2747991	NetLabel: protocols = UNLABELED CIPSOv4
[	0.2748571	NetLabel: unlabeled traffic allowed by default
[	0.275113]	HPET: 3 timers in total, 0 timers will be used for per-cpu timer
[	0.275139]	hpet0: at MMIO 0xfed00000, IRQs 2, 8, 0
[	0.2751751	hpet0: 3 comparators, 64-bit 100.000000 MHz counter
[	0.2770661	Switching to clocksource hpet
[	0.3027061	AppArmor: AppArmor Filesystem Enabled
[	0.3027581	pnp: PnP ACPI init
[	0.3027991	ACPI: bus type PNP registered
[	0.304590]	pnp: PnP ACPI: found 7 devices
[	0.304608]	ACPI: bus type PNP unregistered
[	0.315054]	NET: Registered protocol family 2
[	0.315570]	TCP established hash table entries: 2048 (order: 3, 32768 bytes)
[	0.3156651	TCP bind hash table entries: 2048 (order: 3, 32768 bytes)
[	0.315726]	TCP: Hash tables configured (established 2048 bind 2048)
[	0.315781]	TCP: reno registered
[	0.315805]	UDP hash table entries: 256 (order: 1, 8192 bytes)
[	0.315849]	UDP-Lite hash table entries: 256 (order: 1, 8192 bytes)
[		NET: Registered protocol family 1
I		pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[		pci 0000:00:01.0: PIIX3: Enabling Passive Release
ſ	0.3161751	pci 0000:00:01.0: Activating ISA DMA hang workarounds
[	0.3163511	Trying to unpack rootfs image as initramfs

Log:

fork(): swapper/0 (PID: 2) => 5 (kthreadd)
fork(): swapper/0 (PID: 2) => 6 (kthreadd)
fork(): swapper/0 (PID: 2) => 7 (kthreadd)
fork(): swapper/0 (PID: 2) => 8 (kthreadd)
fork(): swapper/0 (PID: 2) => 9 (kthreadd)
fork(): swapper/0 (PID: 2) => 10 (kthreadd)
fork(): swapper/0 (PID: 2) => 11 (kthreadd)
fork(): swapper/0 (PID: 2) => 12 (kthreadd)
fork(): swapper/0 (PID: 2) => 13 (kthreadd)
fork(): swapper/0 (PID: 2) => 14 (kthreadd)
fork(): swapper/0 (PID: 2) => 15 (kthreadd)
fork(): swapper/0 (PID: 2) => 16 (kthreadd)
fork(): swapper/0 (PID: 2) => 17 (kthreadd)
fork(): swapper/0 (PID: 2) => 18 (kthreadd)
fork(): swapper/0 (PID: 2) => 19 (kthreadd)
fork(): swapper/0 (PID: 2) => 20 (kthreadd)
fork(): swapper/0 (PID: 2) => 21 (kthreadd)
fork(): swapper/0 (PID: 2) => 22 (kthreadd)

Physical Memory Content: \_\_\_\_\_ The second second second and the second ..... . the second s ------

 CycleCount:
 1,582,727,143

 RealTime:
 00:01:18.036

 SimulationTime:
 00:00:03.141

 Writes:
 326,796,211

 1 Byte Writes:
 70,790,401 (21%)

 2 Byte Writes:
 13,868,141 (4%)

 4 Byte Writes:
 68,339,895 (20%)

 8 Byte Writes:
 173,797,774 (53%)

Reads:	402,192,802
Written Data:	1,762,268,455 B
Processed Data:	20,343,675,216 B
Total Write Entries:	27,858,951,214
Total Read Entries:	47,602,879,275

Simutrace

Timeline:

## Simutrace



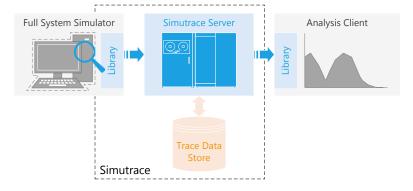
## Demo

Thorsten Gröninger, Marc Rittinghaus, Frank Bellosa – Simutrace

## Conclusion



- Memory tracing puts high pressure on tracing infrastructure
  - Many terabytes of data
  - Billions of entries
- Simutrace: Flexible full system tracing
  - Keeps slowdown at a minimum
  - Delivers high compression
  - Eases access to recorded data
- Will be available as open source
  - http://simutrace.org



### Simutrace makes memory tracing efficient



## Simutrace in Research



- Student projects at KIT
  - Characteristics of memory duplication
  - Applicability of more accurate page access information
- Analysis of memory duplication on NUMA systems

### SimuBoost



- Accelerate full system simulation through massive parallelization
- Significantly increases requirements for tracing infrastructure
  - > 50 simulations in parallel