TUHH
Technische Universität Hamburg

Leibniz Universität Hannover

MELF: Multivariant Executables for a Heterogeneous World

USENIX ATC'23

Dominik Töllner Christian Dietrich Illia Ostapyshyn Florian Rommel Daniel Lohmann

Leibniz Universität Hannover
toellner@sra.uni-hannover.de

2023-07-10

Motivation

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

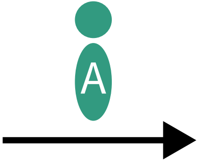
Motivation

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

It's slow!
Tell me why!

Motivation

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```



It's slow!
Tell me why!

-pg profiling

```
validation_t validate_query(
  client_t *c, query_t *q){
    mcount();// Instrumentation
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Motivation

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

A

It's slow!
Tell me why!

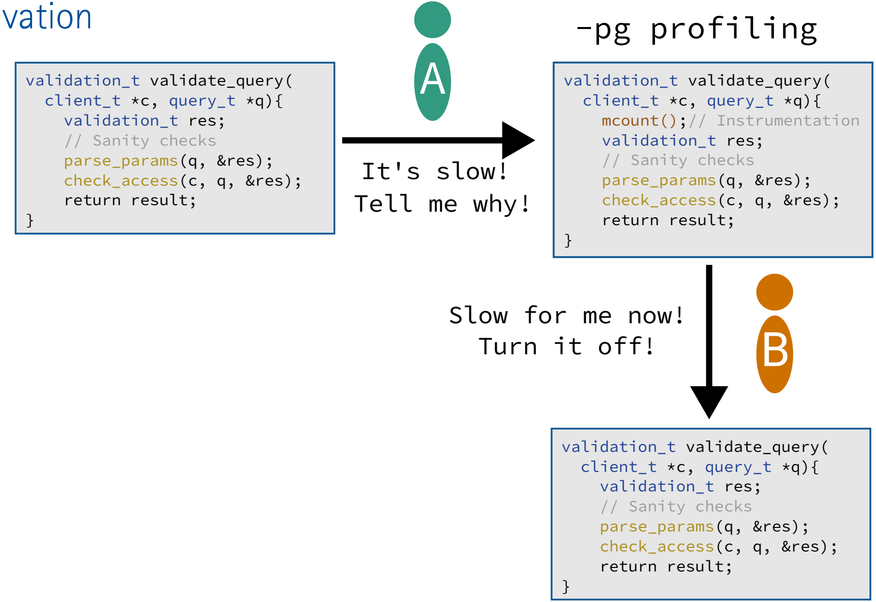
-pg profiling

```
validation_t validate_query(
  client_t *c, query_t *q){
    mcount();// Instrumentation
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Slow for me now!
Turn it off!

B

Motivation



-pg profiling

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

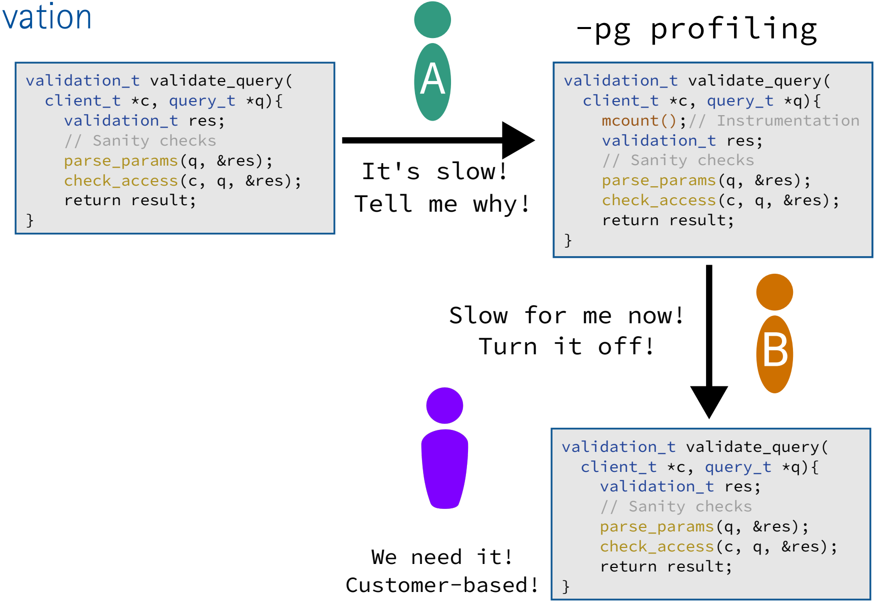
It's slow!
Tell me why!

```
validation_t validate_query(
  client_t *c, query_t *q){
    mcount();// Instrumentation
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Slow for me now!
Turn it off!

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Motivation



```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

It's slow!
Tell me why!

-pg profiling

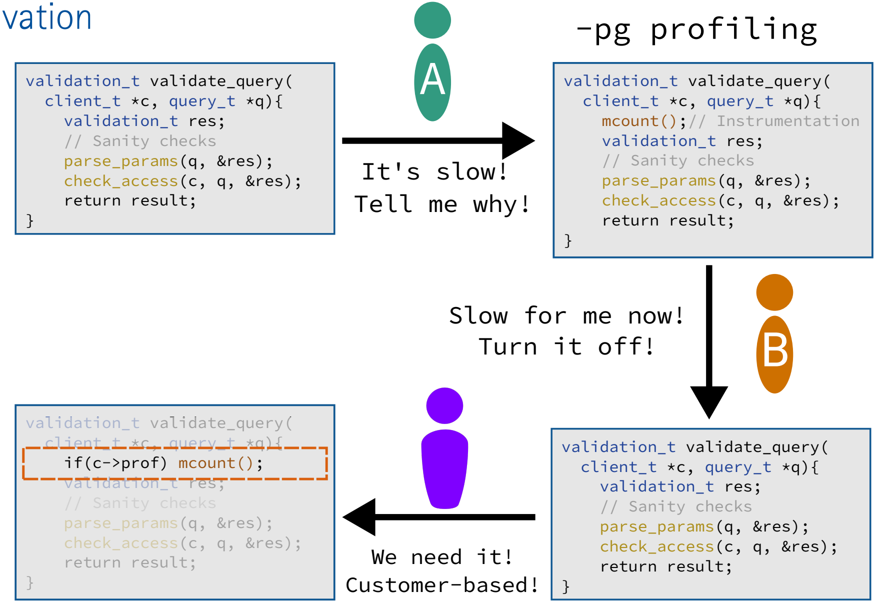
```
validation_t validate_query(
  client_t *c, query_t *q){
    mcount();// Instrumentation
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Slow for me now!
Turn it off!

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

We need it!
Customer-based!

Motivation



-pg profiling

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

It's slow!
Tell me why!

```
validation_t validate_query(
  client_t *c, query_t *q){
    mcount();// Instrumentation
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Slow for me now!
Turn it off!

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

We need it!
Customer-based!

```
validation_t validate_query(
  client_t *c, query_t *q){
    if(c->prof) mcount();
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Motivation

```
validation_t validate_query(
  client_t *c, query_t *q){
    if(c->prof) mcount();
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Motivation

```
validation_t validate_query(
  client_t *c, query_t *q){
    if(c->prof) mcount();
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Checks and Condition Propagation

Motivation

```
validation_t validate_query(
  client_t *c, query_t *q){
    if(c->prof) mcount();
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

Checks and Condition Propagation

Multivariant ELF

Multivariant ELFs

- **Static:** Multivariant Binary Format
  - Multiple compile-time variants of program
  - Function granularity
- **Dynamic:** Overlay Manager
  - Concurrent usage of multiple variants
  - Kernel extension for synchronized address spaces
- **Case-Studies**
  - Performance isoliation of profiling (memcached)
  - Dynamically selectable assertions (sqlite)
  - Context-specific address sanitizer (mariadb)
  - Heterogeneous ISA-aware thread pool

Multivariant ELFs

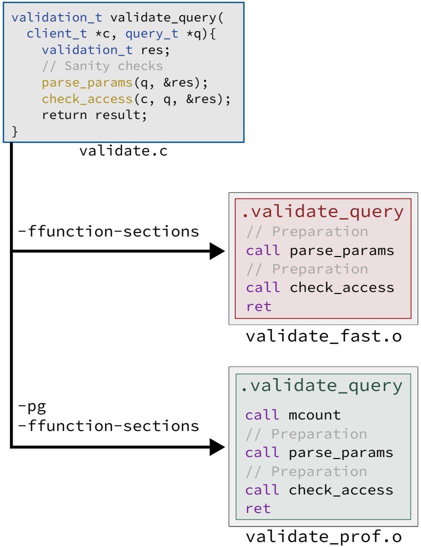
- **Static:** Multivariant Binary Format
  - Multiple compile-time variants of program
  - Function granularity
- **Dynamic:** Overlay Manager
  - Concurrent usage of multiple variants
  - Kernel extension for synchronized address spaces
- **Case-Studies**
  - **Performance isoliation of profiling (memcached)**
  - Dynamically selectable assertions (sqlite)
  - Context-specific address sanitizer (mariadb)
  - **Heterogeneous ISA-aware thread pool**

Building Multivariant ELFs

```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

validate.c

Building Multivariant ELFs



```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

validate.c

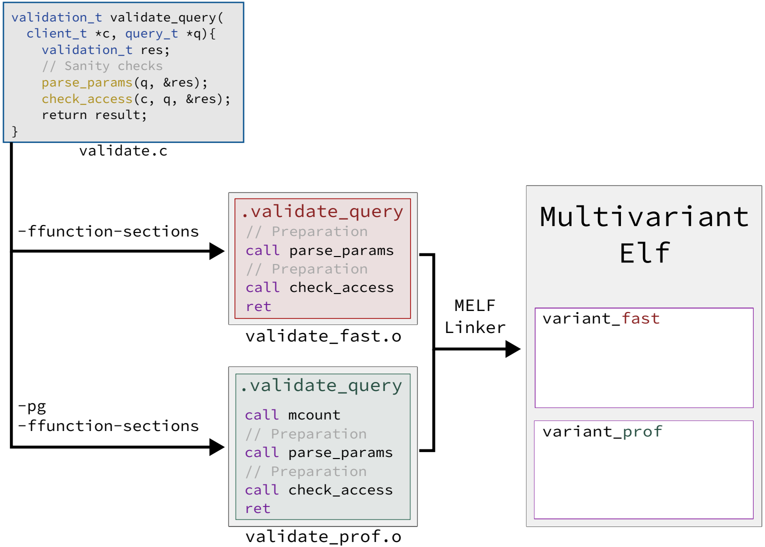
```
.validate_query
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_fast.o

```
.validate_query

call mcount
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_prof.o

Building Multivariant ELFs



```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

validate.c

-ffunction-sections

```
.validate_query
// Preparation
call parse_params
// Preparation
call check_access
ret
```

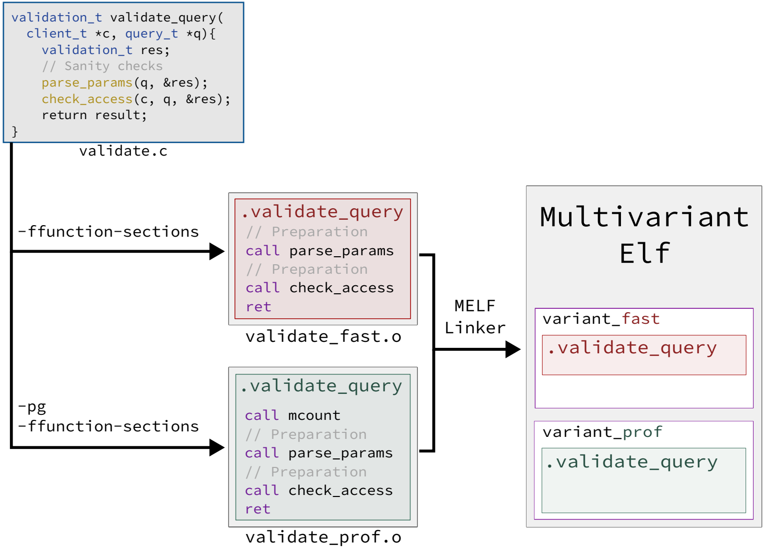
validate_fast.o

-pg
-ffunction-sections

```
.validate_query

call mcount
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_prof.o

Building Multivariant ELFs



```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

validate.c

-ffunction-sections

```
.validate_query
 // Preparation
 call parse_params
 // Preparation
 call check_access
 ret
```

validate_fast.o

-pg
-ffunction-sections

```
.validate_query

 call mcount
 // Preparation
 call parse_params
 // Preparation
 call check_access
 ret
```

validate_prof.o

MELF
Linker

Multivariant
Elf

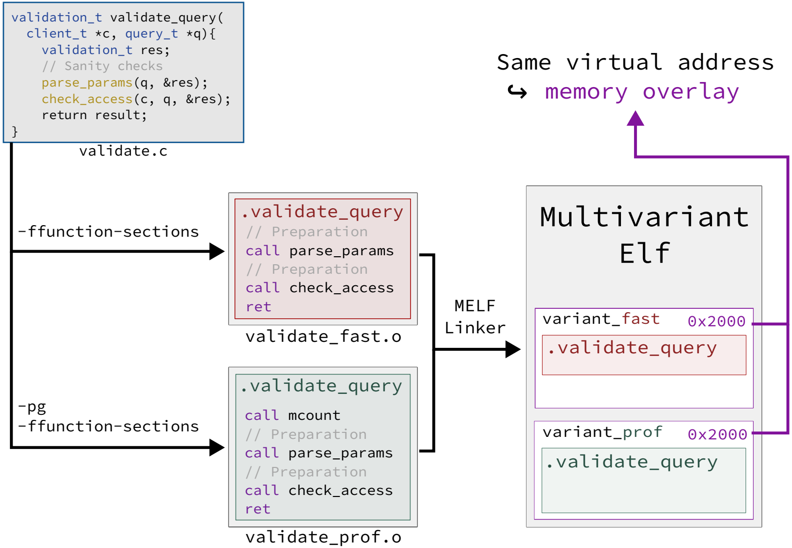
variant_fast

.validate_query

variant_prof

.validate_query

Building Multivariant ELFs



```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

validate.c

-ffunction-sections

```
.validate_query
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_fast.o

-pg
-ffunction-sections

```
.validate_query

call mcount
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_prof.o

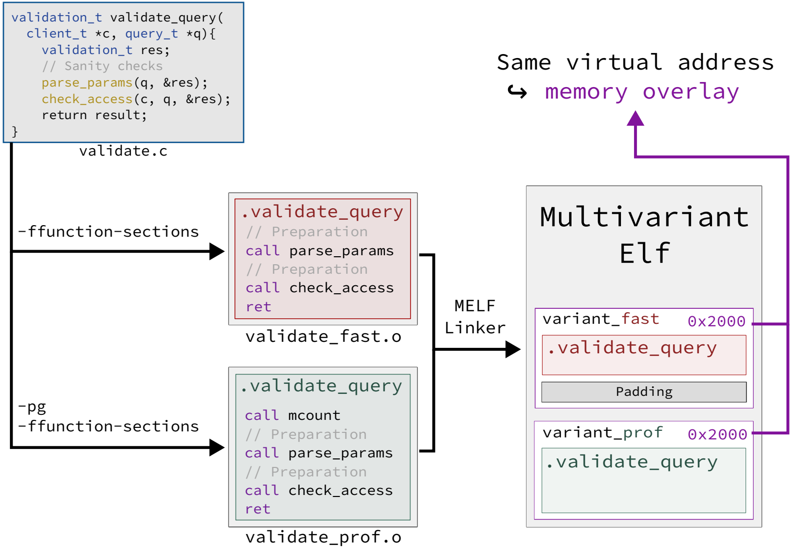
MELF
Linker

Multivariant
Elf

```
variant_fast     0x2000
.validate_query

variant_prof     0x2000
.validate_query
```

Same virtual address
↪ memory overlay

Building Multivariant ELFs



```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

validate.c

-ffunction-sections

```
.validate_query
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_fast.o

-pg
-ffunction-sections

```
.validate_query

call mcount
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_prof.o

MELF Linker

Multivariant Elf

variant_fast 0x2000

.validate_query

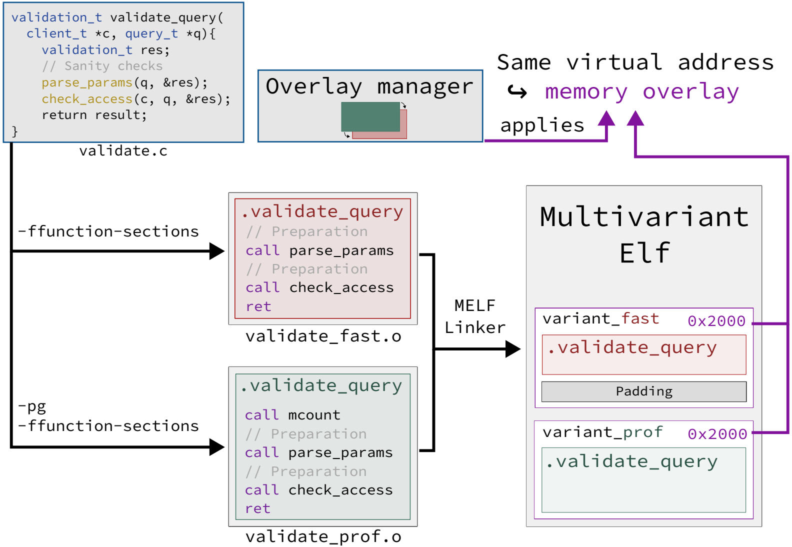
Padding

variant_prof 0x2000

.validate_query

Same virtual address
↪ memory overlay

Building Multivariant ELFs



```
validation_t validate_query(
  client_t *c, query_t *q){
    validation_t res;
    // Sanity checks
    parse_params(q, &res);
    check_access(c, q, &res);
    return result;
}
```

validate.c

```
.validate_query
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_fast.o

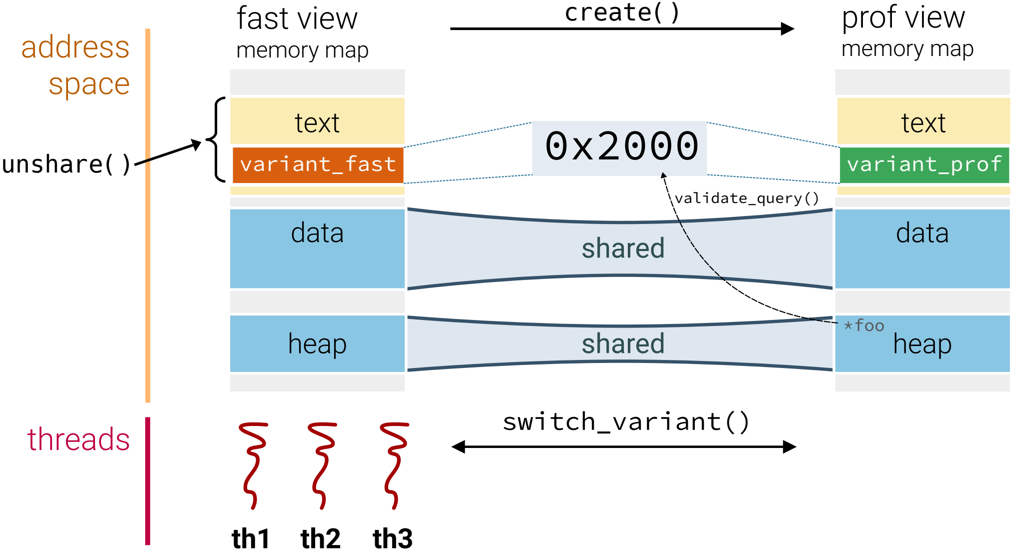
```
.validate_query

call mcount
// Preparation
call parse_params
// Preparation
call check_access
ret
```

validate_prof.o

Same virtual address
↪ memory overlay

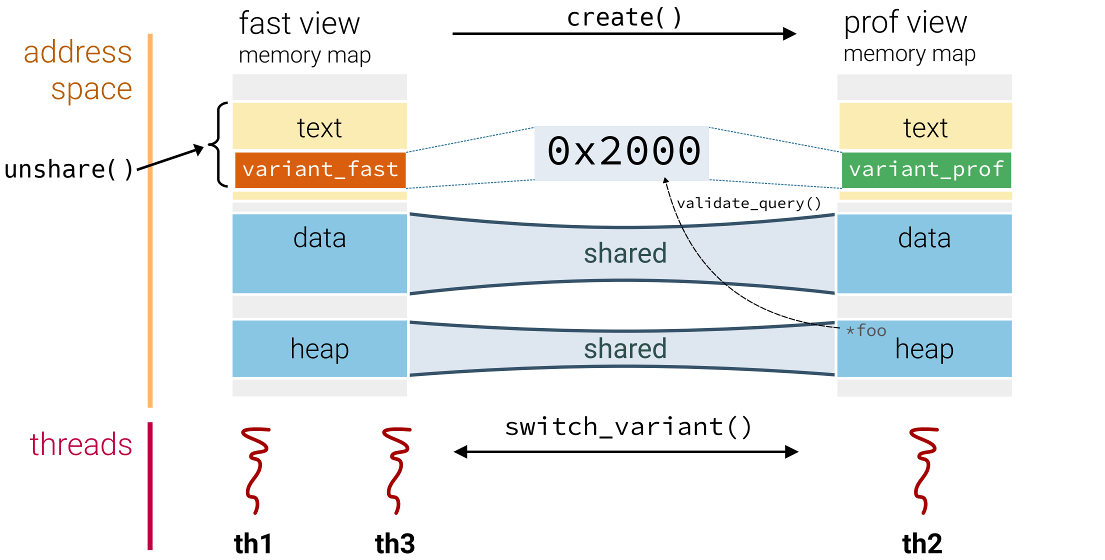
Synchronized Address Space Views

Rommel et al. 2020



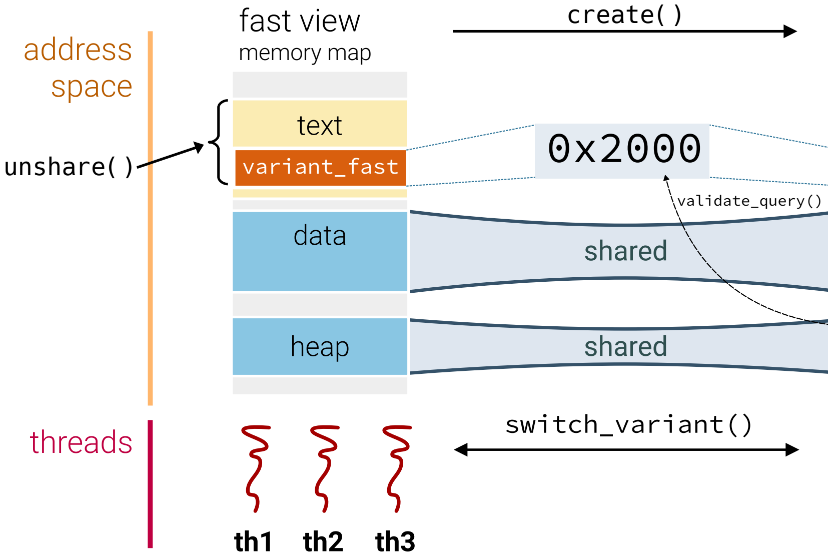
Synchronized Address Space Views

Rommel et al. 2020

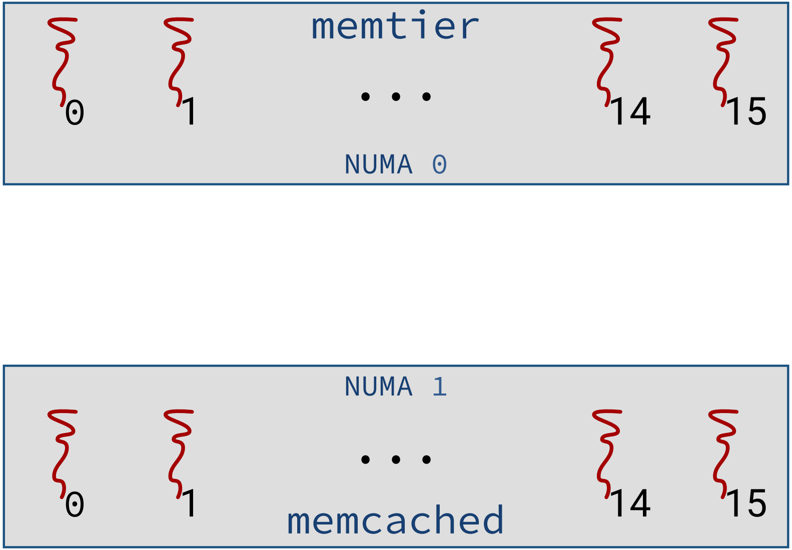


Synchronized Address Space Views

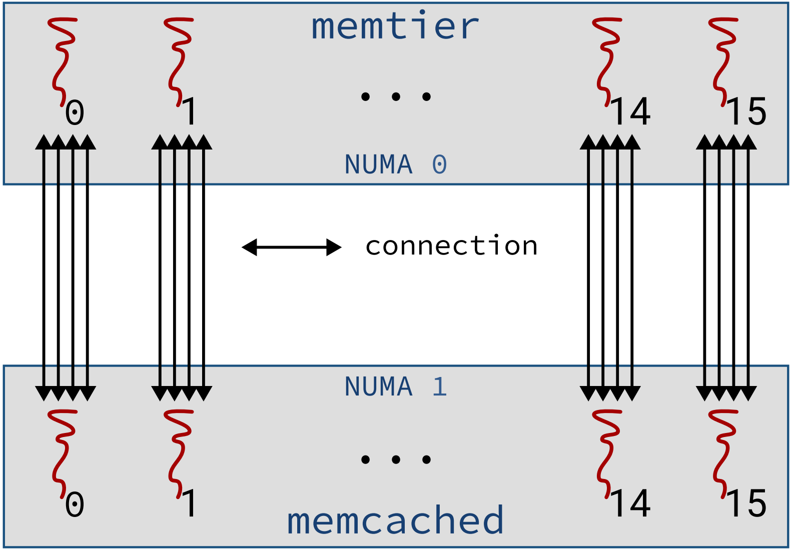
Rommel et al. 2020



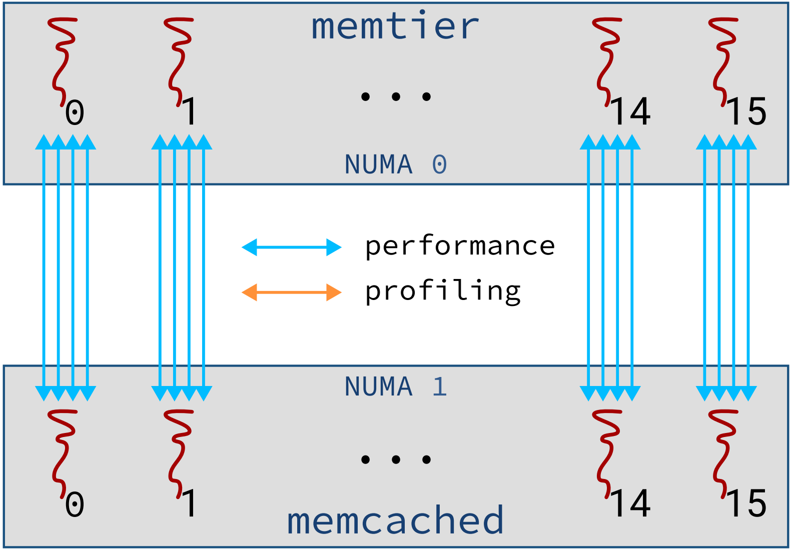
Case-Study: Performance isolation of profiling



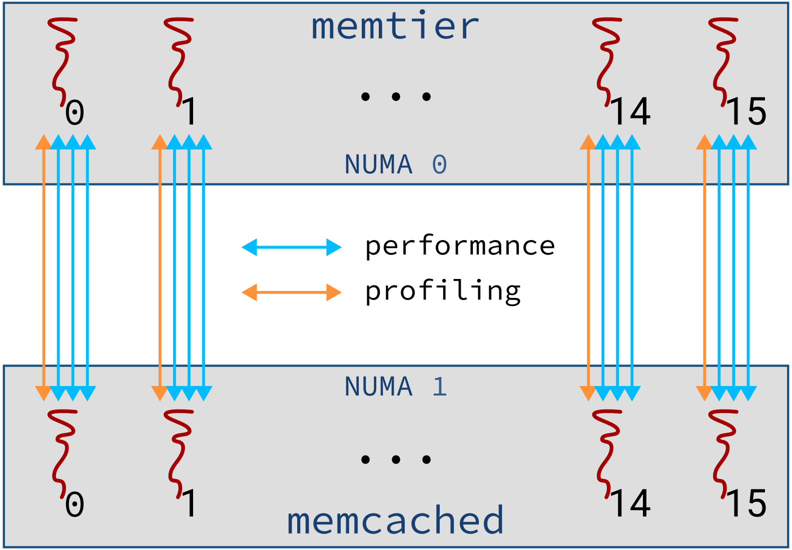
Case-Study: Performance isolation of profiling



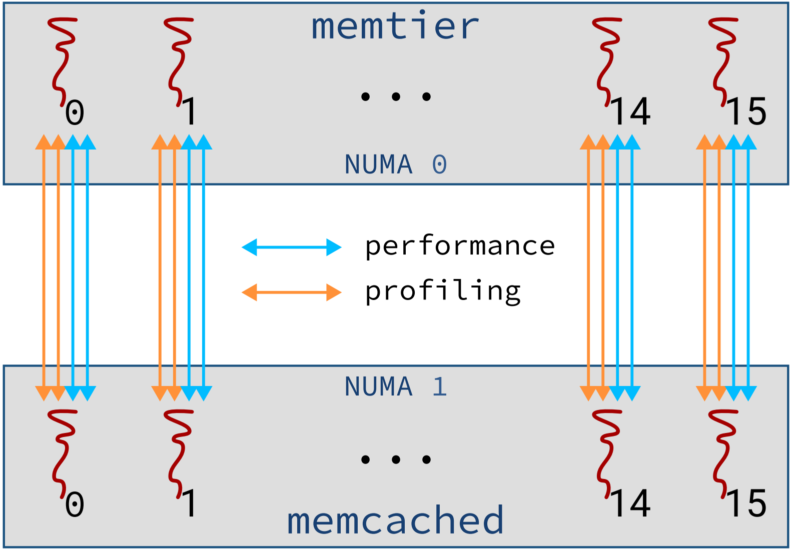
Case-Study: Performance isolation of profiling



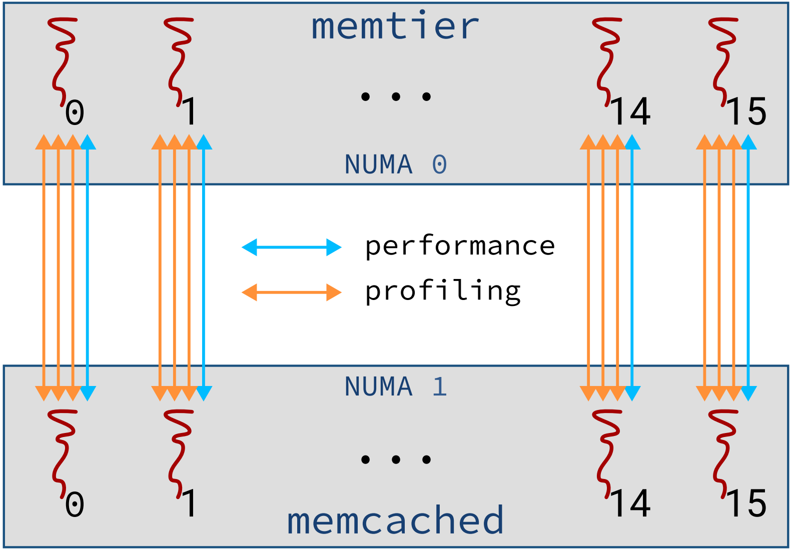
Case-Study: Performance isolation of profiling



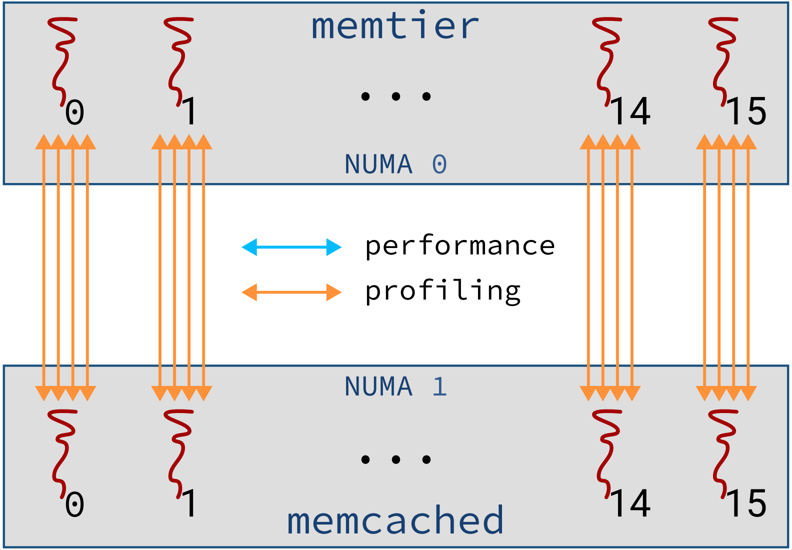
Case-Study: Performance isolation of profiling



Case-Study: Performance isolation of profiling



Case-Study: Performance isolation of profiling

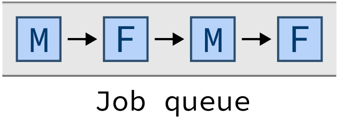


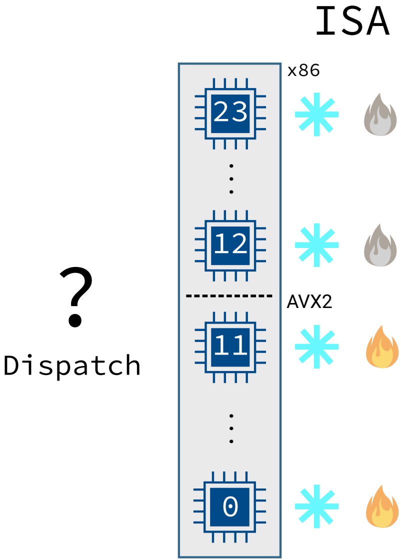
Case-Study: Performance isolation of profiling results

- Latency impact of profiling connections on performance connections
  - 0% profiled connections →0.11% higher mean latency
  - 75% profiled connections →5.94% higher mean latency
- Without MELF: Always activating profiling →175% higher mean latency!

Case-Study: Heterogeneous ISA-aware thread pool

	F	M
✱	57.9ms	58.3ms
🔥	57.9ms	38ms

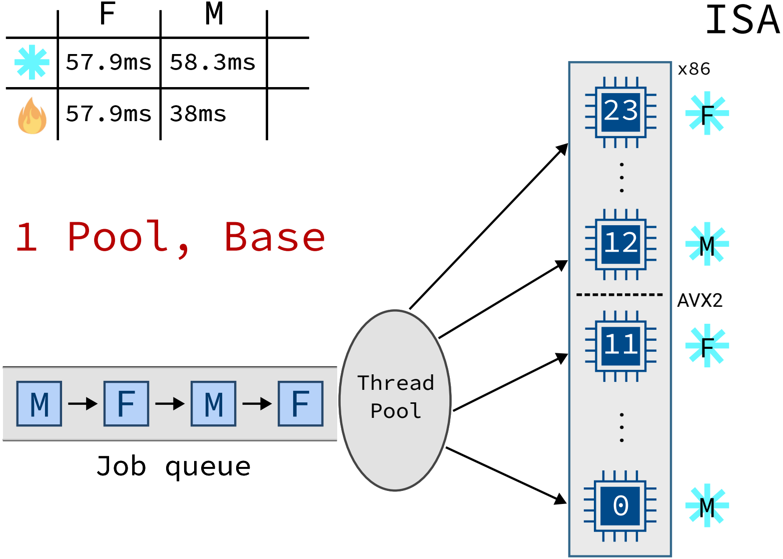




Case-Study: Heterogeneous ISA-aware thread pool

	F	M
✳	57.9ms	58.3ms
🔥	57.9ms	38ms

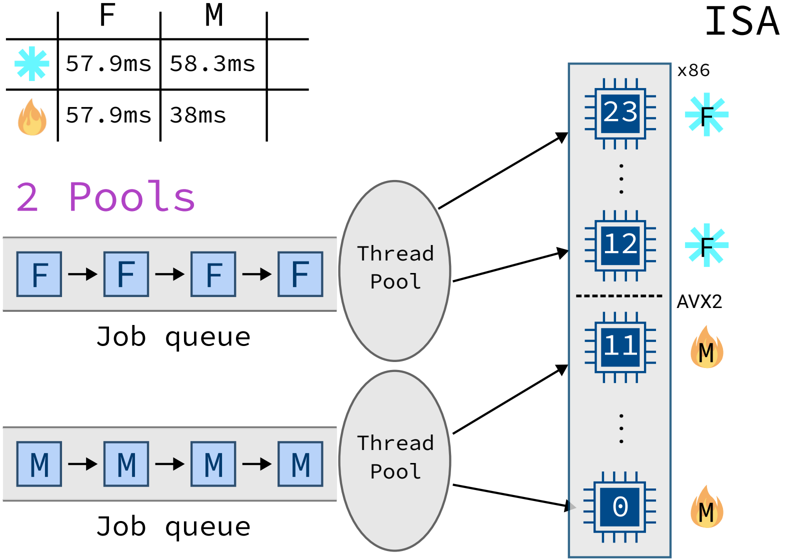
1 Pool, Base



Case-Study: Heterogeneous ISA-aware thread pool

	F	M
✳	57.9ms	58.3ms
🔥	57.9ms	38ms

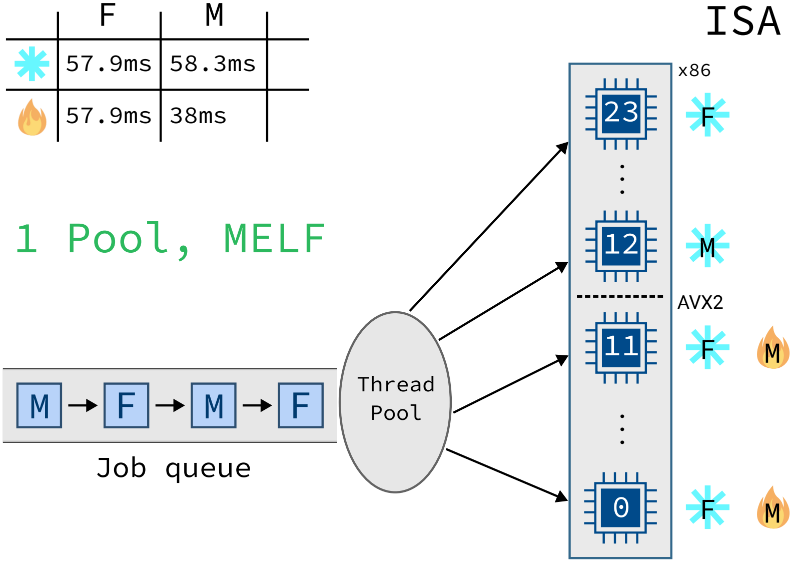
2 Pools



Case-Study: Heterogeneous ISA-aware thread pool

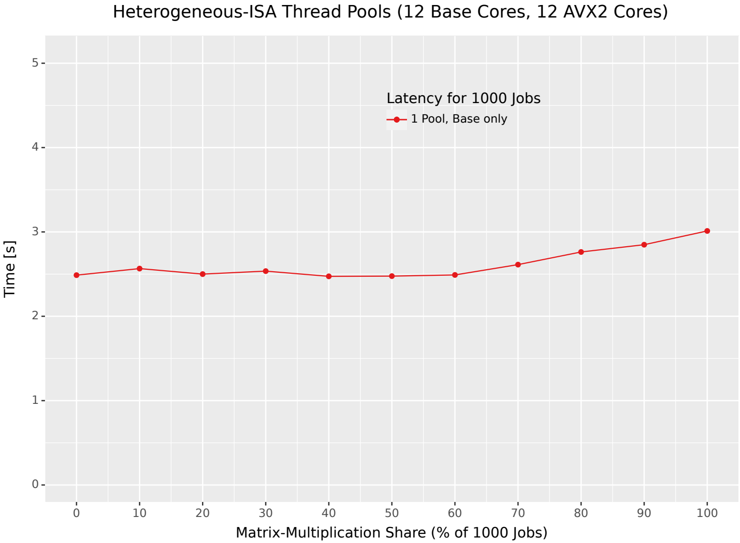
	F	M
✱	57.9ms	58.3ms
🔥	57.9ms	38ms

1 Pool, MELF



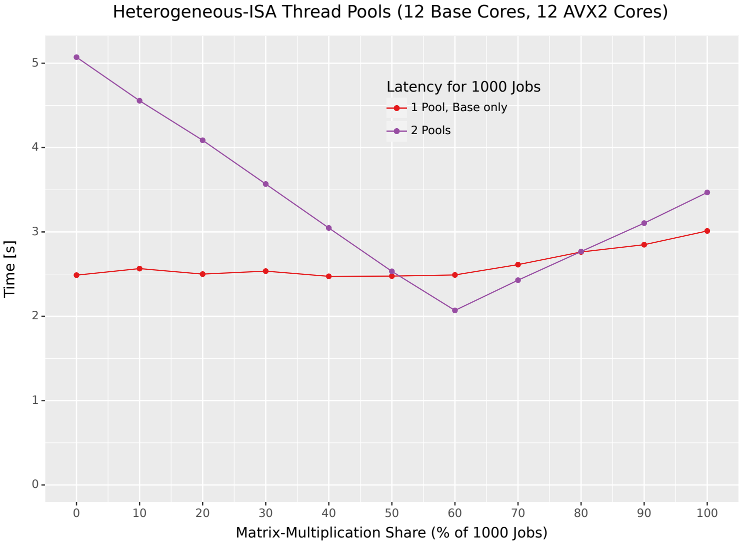
Case-Study: Heterogeneous ISA-aware thread pool results

	F	M	
✱	57.9ms	58.3ms	
🔥	57.9ms	38ms	



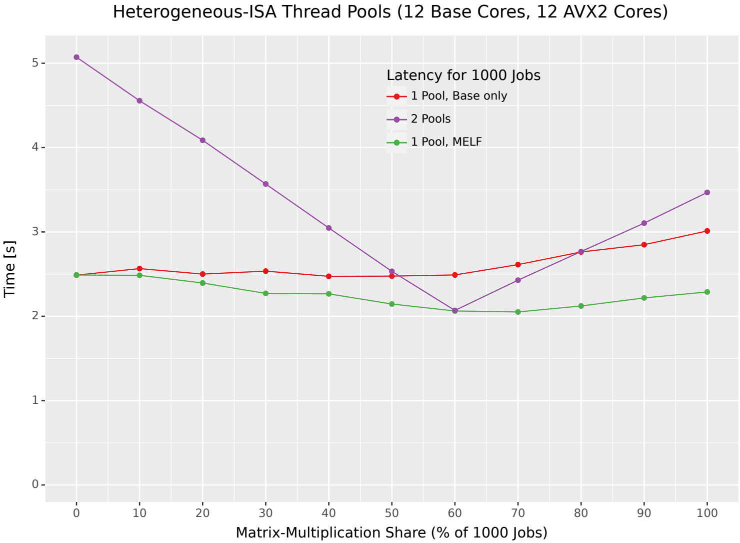
Case-Study: Heterogeneous ISA-aware thread pool results

	F	M
✱	57.9ms	58.3ms
🔥	57.9ms	38ms


Heterogeneous-ISA Thread Pools (12 Base Cores, 12 AVX2 Cores)

Case-Study: Heterogeneous ISA-aware thread pool results

	F	M	
✱	57.9ms	58.3ms	
🔥	57.9ms	38ms	



Conclusion

- MELF: Dynamic exchange of compile-time variants on a per-function level via memory overlaying
- Thread-local adaptation via address space views
- Performance isoliation of profiling (memcached)
- Heterogeneous ISA-aware thread pool
- Dynamically selectable assertions (sqlite)
- Context-specific address sanitizer (mariadb)

Varability mechanism for a heterogeneous world

ARTIFACT EVALUATED
usenix ASSOCIATION
AVAILABLE

ARTIFACT EVALUATED
usenix ASSOCIATION
FUNCTIONAL

ARTIFACT EVALUATED
usenix ASSOCIATION
REPRODUCED

toellner@sra.uni-hannover.de

Dominik Töllner et al. Multivariant Executables for a Heterogeneous World – Conclusion 10 – 10