Testing Side-Channel Security of Cryptographic Implementations against Future Microarchitectures

Chitchanok Chuengsatiansup and Marco Guarnieri

RWC 2025

Joint work with Gilles. Barthe, Marcel Böhme, Sunjay Cauligi,
Daniel Genkin, David Mateos Romero, Peter Schwabe, David Wu, and Yuval Yarom

Motivation

Motivation

- Microarchitectural optimization $\rightarrow$ speed up computation 😊

Motivation

- Microarchitectural optimization $\rightarrow$ speed up computation 😊
- Sanchez Vicarte et al., ISCA 2021
  - identify many security-critical optimization proposals ☹️

Motivation

- Microarchitectural optimization $\rightarrow$ speed up computation 😊
- Sanchez Vicarte et al., ISCA 2021
  - identify many security-critical optimization proposals ☹️
- How to evaluate security impacts?

Motivation

- Microarchitectural optimization $\rightarrow$ speed up computation 😊
- Sanchez Vicarte et al., ISCA 2021
    - identify many security-critical optimization proposals ☹
- How to evaluate security impacts?
    - cryptographic implementations

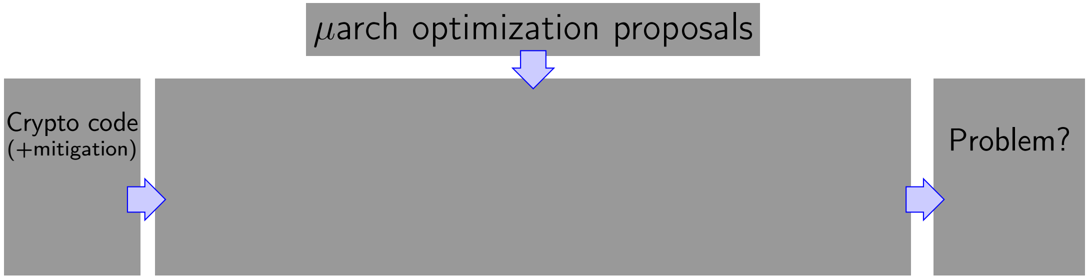
Motivation

- Microarchitectural optimization $\rightarrow$ speed up computation 😊
- Sanchez Vicarte et al., ISCA 2021
  - identify many security-critical optimization proposals ☹️
- How to evaluate security impacts?
  - cryptographic implementations
  - existing mitigations $\rightarrow$ do they really help preventing leaks?

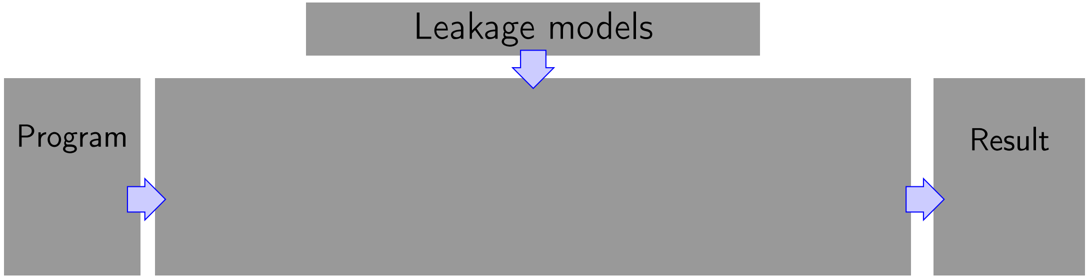
Motivation

- Microarchitectural optimization $\rightarrow$ speed up computation 😊
- Sanchez Vicarte et al., ISCA 2021
  - identify many security-critical optimization proposals ☹
- How to evaluate security impacts?
  - cryptographic implementations
  - existing mitigations $\rightarrow$ do they really help preventing leaks?
  - future microarchitectural optimizations

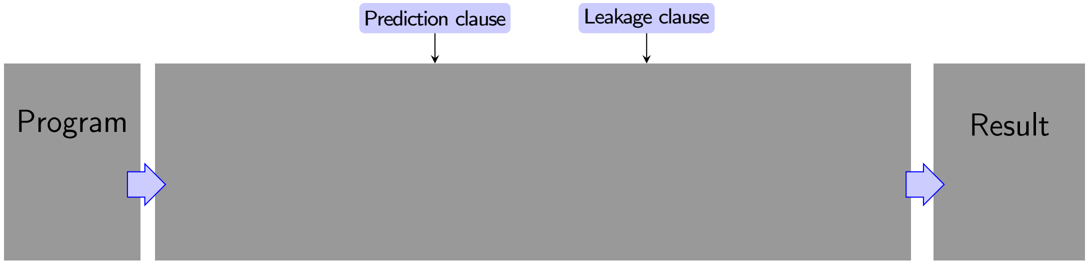
Our framework



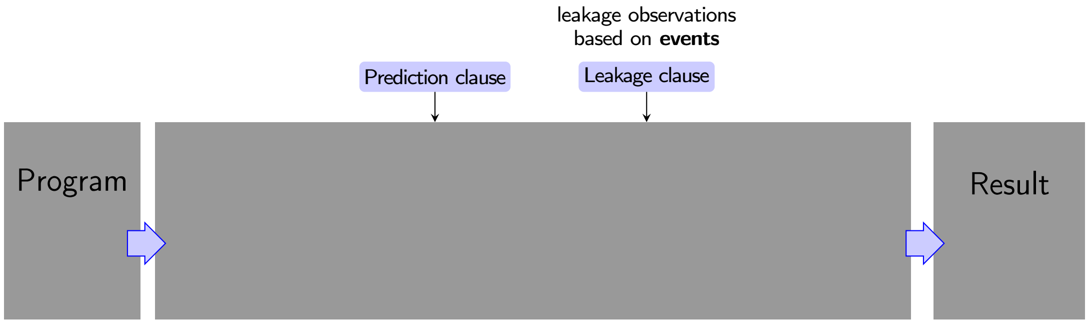
Our framework



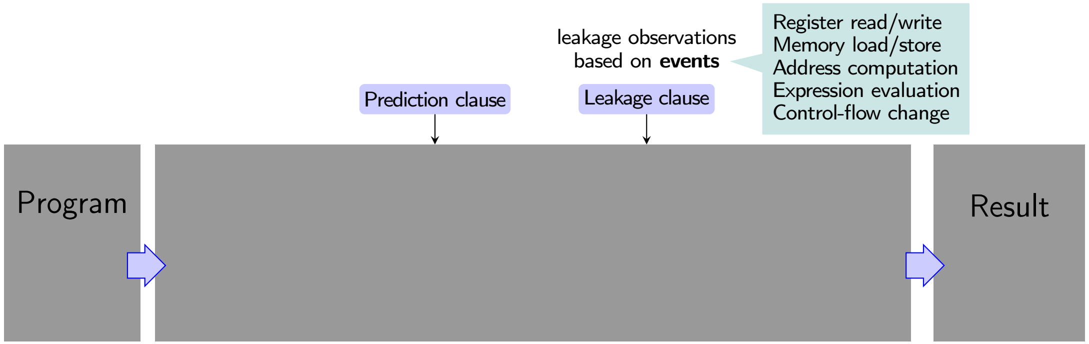
Specify leakage models: LmSpec



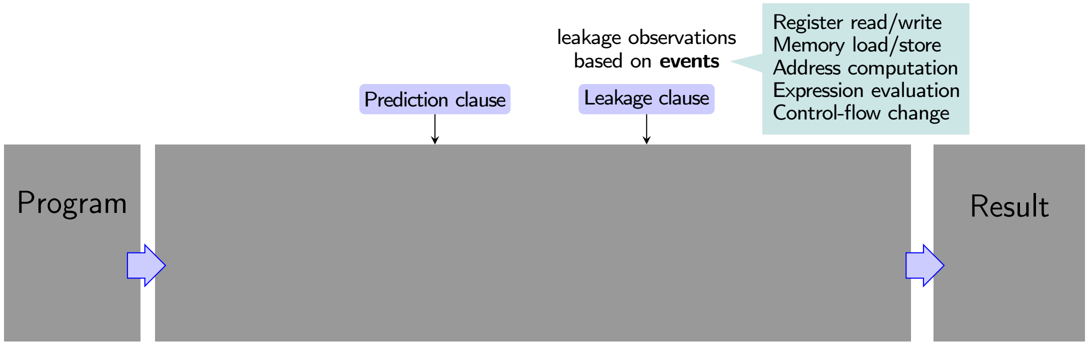
Specify leakage models: LmSpec



Specify leakage models: LMSPEC

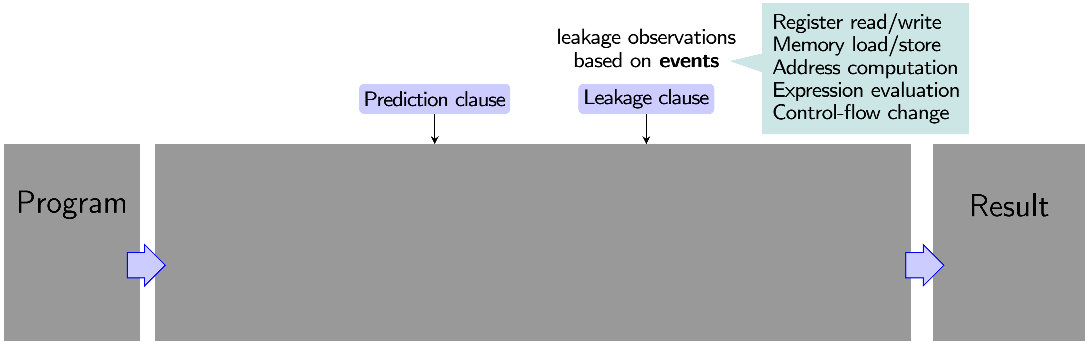


Specify leakage models: LMSPEC



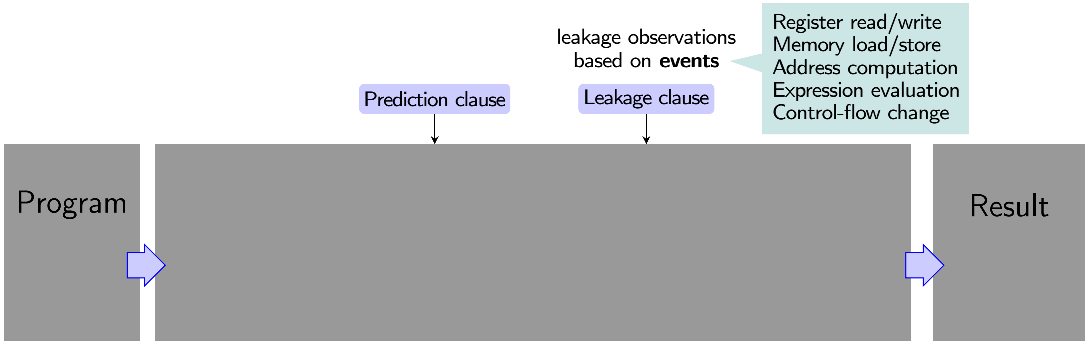
```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

Specify leakage models: LMSPEC



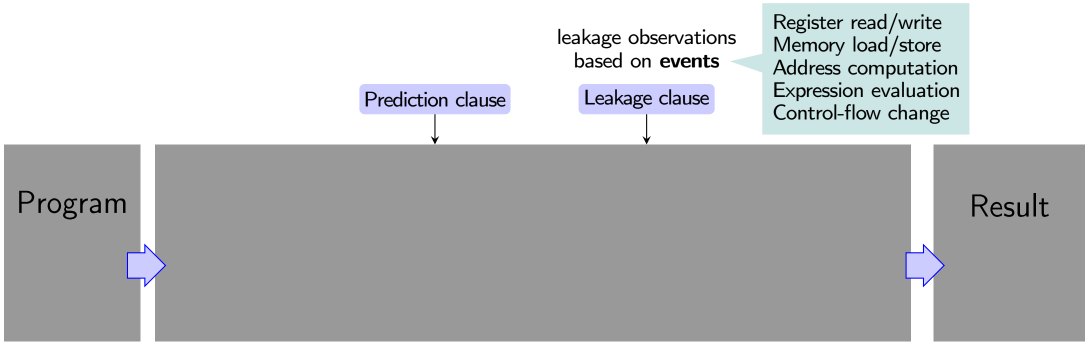
```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

Specify leakage models: LMSPEC



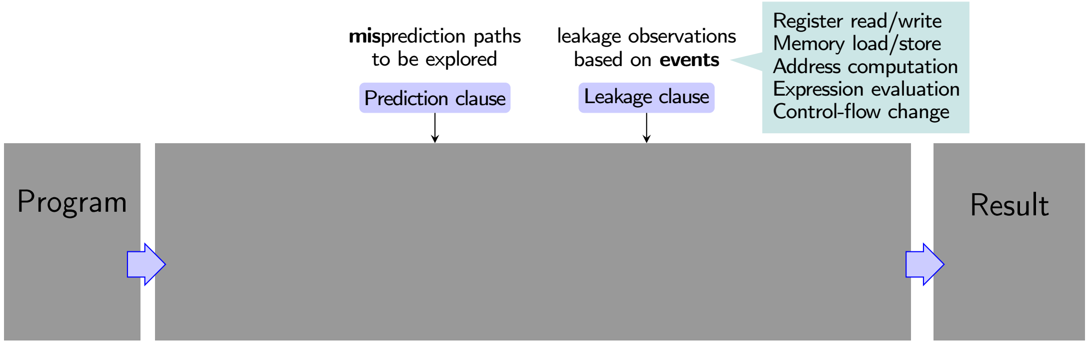
```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
        (when (= val (&mem.read addr sz))
          #("ss" addr val))]))
```

Specify leakage models: LmSpec



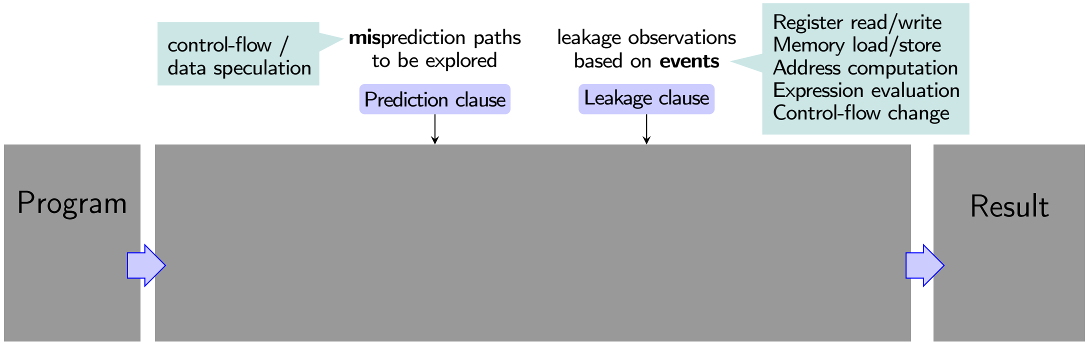
```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
        #("ss" addr val))]))
```

Specify leakage models: LmSpec



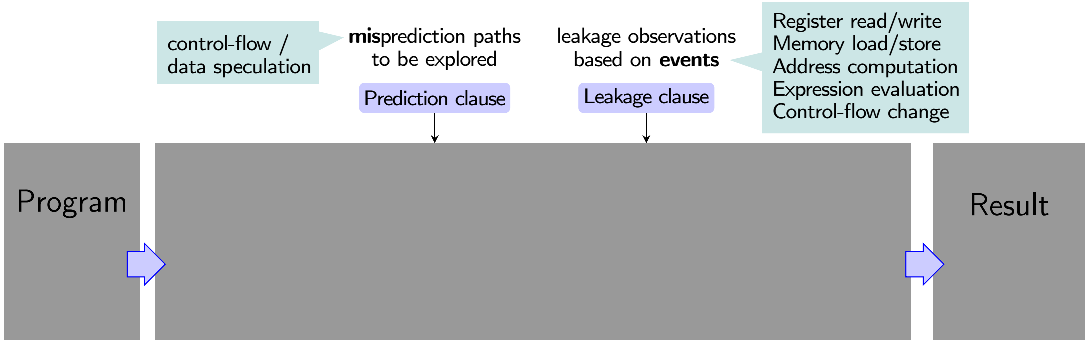
```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

Specify leakage models: LMSPEC



```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

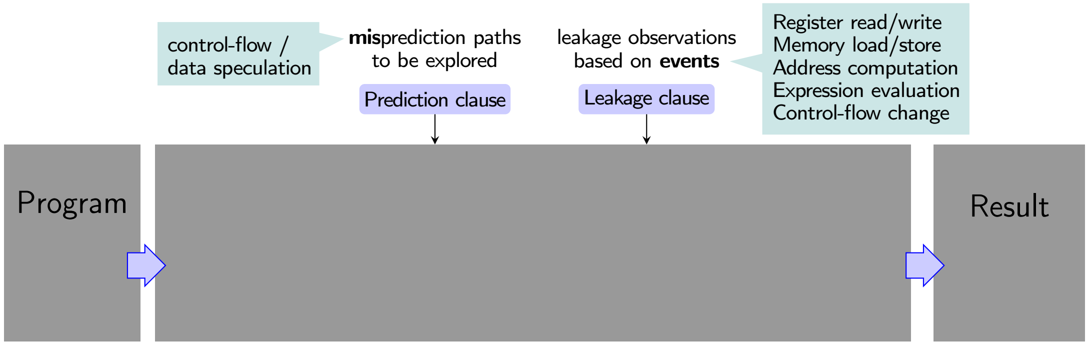
Specify leakage models: LmSpec



```
(defpredictor branchSpec []
(on [(jump addr : n)
     (when (&insn.group CS_GRP_JUMP)
       [("PC" (if n (+ &pc &insn.size) addr))])]))
```

```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

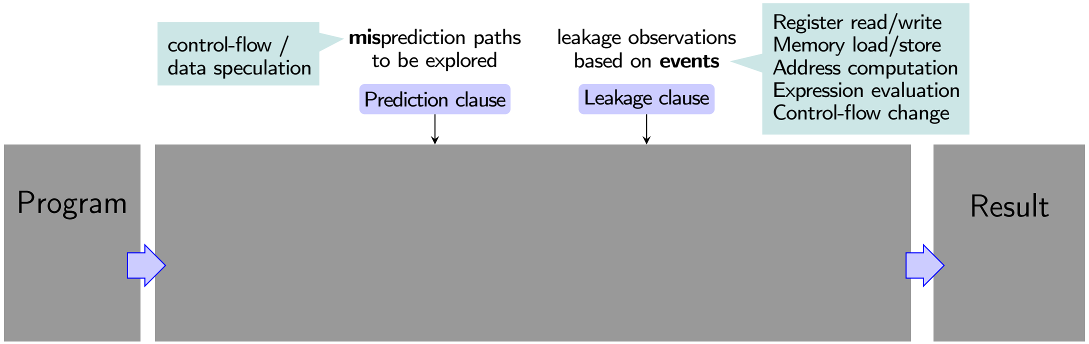
Specify leakage models: LmSpec



```
(defpredictor branchSpec []
(on [(jump addr : n)
     (when (&insn.group CS_GRP_JUMP)
       [("PC" (if n (+ &pc &insn.size) addr))])]))
```

```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

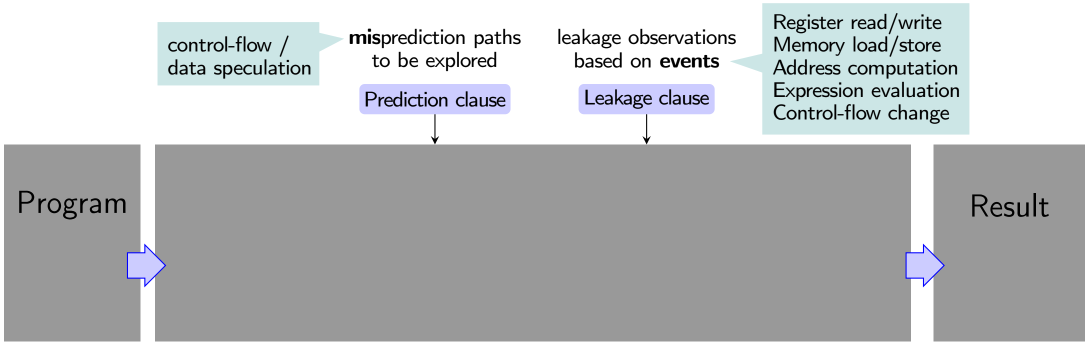
Specify leakage models: LmSpec



```
(defpredictor branchSpec []
(on [(jump addr : n)
    (when (&insn.group CS_GRP_JUMP)
      [("PC" (if n (+ &pc &insn.size) addr))])]))
```

```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

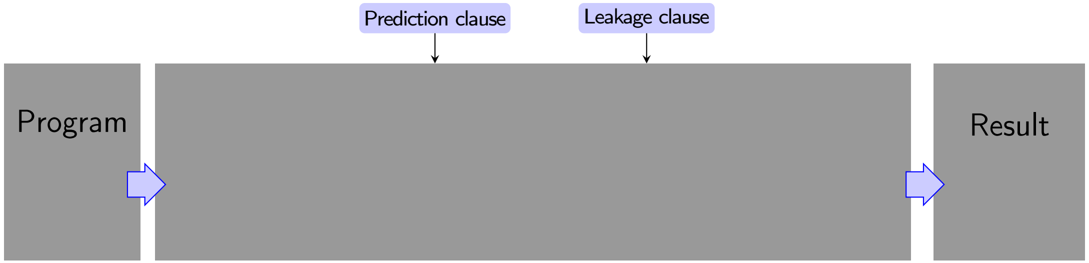
Specify leakage models: LMSPEC



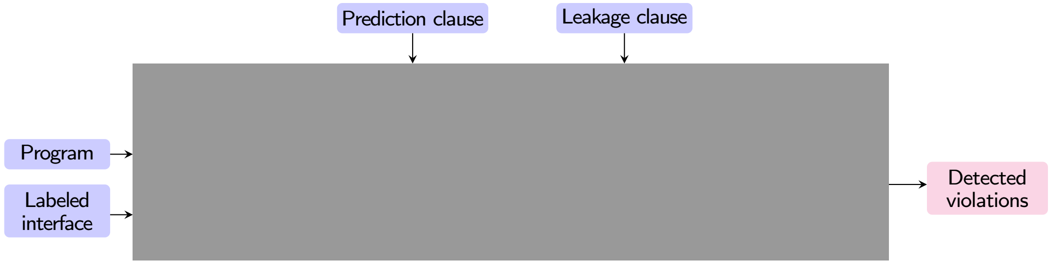
```
(defpredictor branchSpec []
(on [(jump addr : n)
    (when (&insn.group CS_GRP_JUMP)
      [("PC" (if n (+ &pc &insn.size) addr))])]))
```

```
(defleakage SilentStore []
  (on [(store [addr]_sz := val)
       (when (= val (&mem.read addr sz))
         #("ss" addr val))]))
```

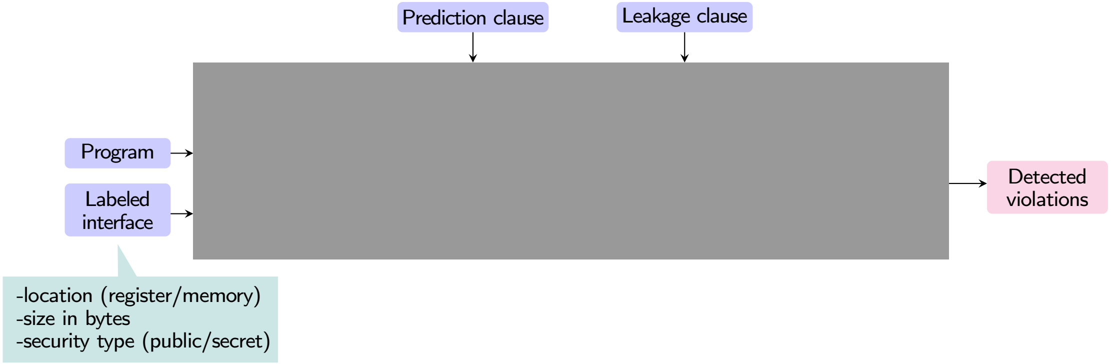
Test for leakage: LmTest



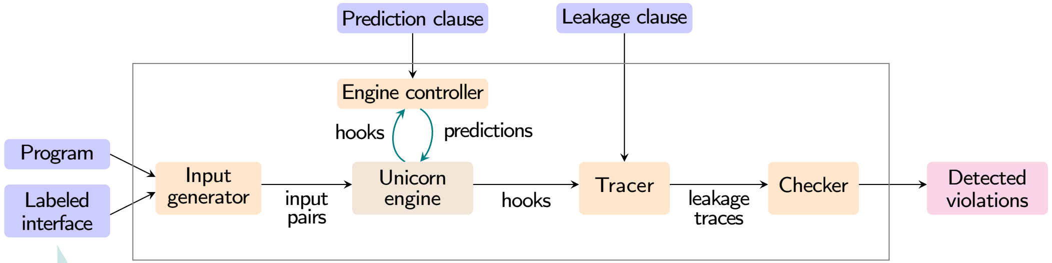
Test for leakage: LmTest



Test for leakage: LMTEST

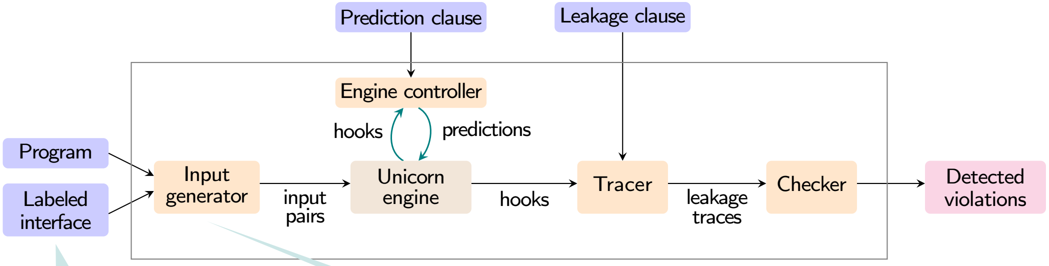


Test for leakage: LmTest



-location (register/memory)
-size in bytes
-security type (public/secret)

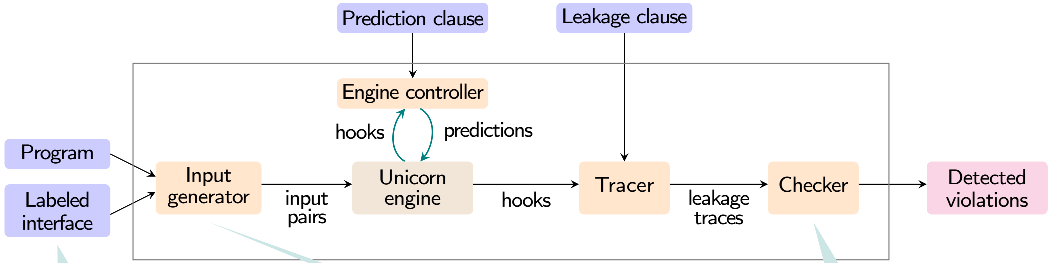
Test for leakage: LMTEST



-location (register/memory)
-size in bytes
-security type (public/secret)

1)generate initial state at random
2)change secret parts to get another state

Test for leakage: LmTest



-location (register/memory)
-size in bytes
-security type (public/secret)

1)generate initial state at random
2)change secret parts to get another state

Leak detected if leakage traces differ

Case study

Case study

- 18 microarchitecture optimization (leakage clause)

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
  - sequential,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line,

Case study

- 18 microarchitecture optimization (leakage clause)
    - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
    - sequential, conditional branch, straight-line, store bypass,

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address (x2)

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address (x2)

108 leakage models

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,**+** their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address (x2)

108 leakage models

- 8 cryptographic primitives

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification, **OP** operand packing, **CR** computation reuse, **CC** cacheline compression, **PF** prefetching, + their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address (x2)

108 leakage models

- 8 cryptographic primitives
  - AES, SHA512, HMAC Stream-XOR, Salsa20, Poly1305, Ed25519, X25519

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,+ their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address (x2)

108 leakage models

- 8 cryptographic primitives
  - AES, SHA512, HMAC Stream-XOR, Salsa20, Poly1305, Ed25519, X25519
- 5 popular libraries

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,+ their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address (x2)

108 leakage models

- 8 cryptographic primitives
  - AES, SHA512, HMAC Stream-XOR, Salsa20, Poly1305, Ed25519, X25519
- 5 popular libraries
  - libsodium, libjade, libnettle, cryptlib, rust-crypto

Case study

- 18 microarchitecture optimization (leakage clause)
  - **CT** constant time, **SS** silent store, **RFC** register file compression, **CS** computation simplification,**OP** operand packing,**CR** computation reuse, **CC** cacheline compression,**PF** prefetching,+ their variations
- 6 execution models (prediction clause)
  - sequential, conditional branch, straight-line, store bypass, return address (x2)

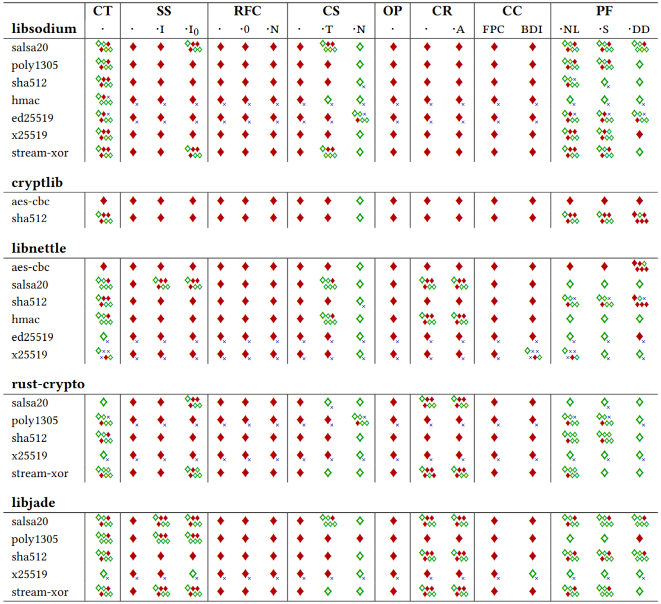
108 leakage models

- 8 cryptographic primitives
  - AES, SHA512, HMAC Stream-XOR, Salsa20, Poly1305, Ed25519, X25519
- 5 popular libraries
  - libsodium, libjade, libnettle, cryptlib, rust-crypto

25 implementations

Evaluation results

Evaluation results



	CT	SS			RFC			CS			OP	CR		CC		PF		
libsodium	·	·	·I	·I_0	·	·0	·N	·	·T	·N	·	·	·A	FPC	BDI	·NL	·S	·DD
salsa20	◇◇◆/◆◇◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◆◆/◇◇◇	◇	◆	◆	◆	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇◇◆/◇◇◇
poly1305	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇×/◆◇◇	◇×	◇
hmac	◇◆×/◇◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
ed25519	◇◆×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◆×/◆◇◇	◇◆×/◆◇◇	◇◇×/◆◇◇
x25519	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◆◇/◆◇◇	◆
stream-xor	◇◆◆/◆◇◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◆◆/◇◇◇	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◇◆/◆◇◇	◇
cryptlib																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆◇◆/◆◆◆
libnettle																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆◆◇/◆◆◆
salsa20	◇◇◆/◇◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇	◇
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇×/◆◇◇	◇◇×/◆◇◇	◆◇×/◆◆◆
hmac	◇◇◆/◇◇◇	◆	◆	◆	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇	◇
ed25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×
x25519	◇××/×◆◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇××/×◆◇	◇××/×◆◇	◇×	◇×
rust-crypto																		
salsa20	◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇×	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇×	◇
poly1305	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◇◇×/◆◇◇	◇×
sha512	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◇/◆◇◇	◇◇◇/◆◇◇	◇
x25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
stream-xor	◇◇◇/◆◇◇	◆	◆	◇◆◇/◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆/◆◇◆	◇◆◆/◆◇◇	◆	◆	◇◇◇/◆◇◇	◇	◇
libjade																		
salsa20	◇◇◆/◆◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◇◇◇	◇◇◆/◇◇◇
poly1305	◇◇◆/◇◇◇	◆	◇◆◆/◇◇◇	◇◆◆/◇◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◇	◆
sha512	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇◇◆/◇◇◇
x25519	◇×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇×	◇×	◇×	◇×
stream-xor	◇◇◆/◆◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◇◇◇	◇

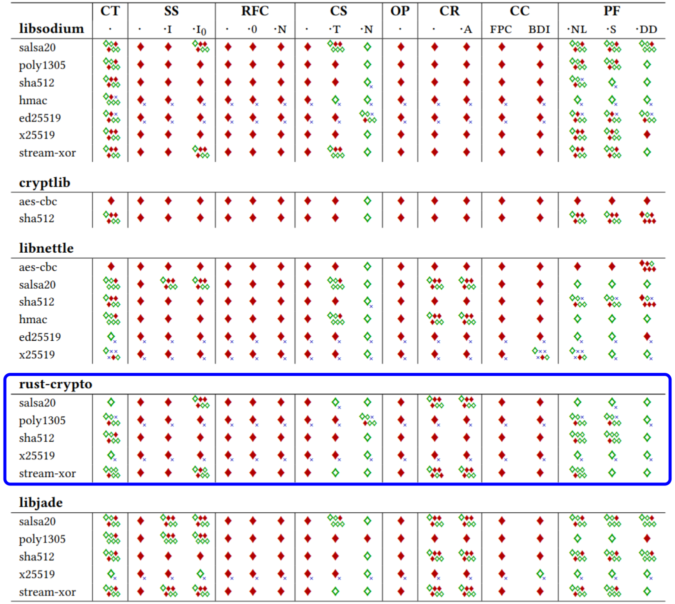
Evaluation results

- All analyzed implementations leak

	CT	SS			RFC			CS			OP	CR		CC		PF		
libsodium	·	·	·I	·I_0	·	·0	·N	·	·T	·N	·	·	·A	FPC	BDI	·NL	·S	·DD
salsa20	◇◇♦ ♦◇◇	♦	♦	◇♦♦ ♦◇◇	♦	♦	♦	♦	◇♦♦ ◇◇◇	◇	♦	♦	♦	♦	♦	◇◇♦ ♦◇◇	◇◇♦ ♦◇◇	◇◇♦ ◇◇◇
poly1305	◇◇♦ ♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇◇♦ ♦◇◇	◇◇♦ ♦◇◇	◇
sha512	◇♦♦ ♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇×	♦	♦	♦	♦	♦	◇◇× ♦◇◇	◇×	◇
hmac	◇♦× ◇◇◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	◇×
ed25519	◇♦× ♦◇◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇◇× ♦◇◇	♦×	♦×	♦×	♦×	♦×	◇♦× ♦◇◇	◇♦× ♦◇◇	◇◇× ♦◇◇
x25519	◇♦♦ ♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇♦♦ ♦◇◇	◇♦◇ ♦◇◇	♦
stream-xor	◇♦♦ ♦◇◇	♦	♦	◇♦♦ ♦◇◇	♦	♦	♦	♦	◇♦♦ ◇◇◇	◇	♦	♦	♦	♦	♦	◇♦♦ ♦◇◇	◇◇♦ ♦◇◇	◇
cryptlib																		
aes-cbc	♦	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	♦	♦	♦
sha512	◇♦♦ ♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦◇♦ ♦♦♦
libnettle																		
aes-cbc	♦	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	♦	♦	♦♦◇ ♦♦♦
salsa20	◇◇♦ ◇◇◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	♦	♦	◇◇♦ ◇◇◇	◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	◇	◇	◇
sha512	◇♦♦ ♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇×	♦	♦	♦	♦	♦	◇◇× ♦◇◇	◇◇× ♦◇◇	♦◇× ♦♦♦
hmac	◇◇♦ ◇◇◇	♦	♦	♦	♦	♦	♦	♦	◇◇♦ ◇◇◇	◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	◇	◇	◇
ed25519	◇×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	♦×
x25519	◇×× ×♦◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	◇×× ×♦◇	◇×× ×♦◇	◇×	◇×
rust-crypto																		
salsa20	◇	♦	♦	◇♦♦ ♦◇◇	♦	♦	♦	♦	◇×	◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	◇	◇×	◇
poly1305	◇◇× ♦◇◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇◇× ♦◇◇	♦×	♦×	♦×	♦×	♦×	◇◇× ♦◇◇	◇◇× ♦◇◇	◇×
sha512	◇◇♦ ♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇◇◇ ♦◇◇	◇◇◇ ♦◇◇	◇
x25519	◇×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	◇×
stream-xor	◇◇◇ ♦◇◇	♦	♦	◇♦◇ ♦◇◇	♦	♦	♦	♦	◇	◇	♦	◇♦♦ ♦◇♦	◇♦♦ ♦◇◇	♦	♦	◇◇◇ ♦◇◇	◇	◇
libjade																		
salsa20	◇◇♦ ♦◇◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	♦	♦	◇◇♦ ◇◇◇	◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	◇◇♦ ♦◇◇	◇◇♦ ◇◇◇	◇◇♦ ◇◇◇
poly1305	◇◇♦ ◇◇◇	♦	◇♦♦ ◇◇◇	◇♦♦ ◇◇◇	♦	♦	♦	♦	♦	♦	♦	♦	♦	♦	♦	◇	◇	♦
sha512	◇◇♦ ♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	◇◇♦ ♦◇◇	◇◇♦ ♦◇◇	◇◇♦ ◇◇◇
x25519	◇×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	◇×	◇×	◇×	◇×
stream-xor	◇◇♦ ♦◇◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	♦	♦	◇	◇	♦	◇♦♦ ♦◇◇	◇♦♦ ♦◇◇	♦	♦	◇◇♦ ♦◇◇	◇◇♦ ◇◇◇	◇

Evaluation results

- All analyzed implementations leak
- Memory-safe languages (e.g., Rust) not significantly mitigate leaks



	CT	SS			RFC			CS			OP	CR		CC		PF		
libsodium	·	·	·I	·I_0	·	·0	·N	·	·T	·N	·	·	·A	FPC	BDI	·NL	·S	·DD
salsa20	◇◇◆ ◆◇◇	◆	◆	◇◆◆ ◆◇◇	◆	◆	◆	◆	◇◆◆ ◇◇◇	◇	◆	◆	◆	◆	◆	◇◇◆ ◆◇◇	◇◇◆ ◆◇◇	◇◇◆ ◇◇◇
poly1305	◇◇◆ ◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◆ ◆◇◇	◇◇◆ ◆◇◇	◇
sha512	◇◆◆ ◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇× ◆◇◇	◇×	◇
hmac	◇◆× ◇◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
ed25519	◇◆× ◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇× ◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◆× ◆◇◇	◇◆× ◆◇◇	◇◇× ◆◇◇
x25519	◇◆◆ ◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆ ◆◇◇	◇◆◇ ◆◇◇	◆
stream-xor	◇◆◆ ◆◇◇	◆	◆	◇◆◆ ◆◇◇	◆	◆	◆	◆	◇◆◆ ◇◇◇	◇	◆	◆	◆	◆	◆	◇◆◆ ◆◇◇	◇◇◆ ◆◇◇	◇
cryptlib																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆
sha512	◇◆◆ ◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆◇◆ ◆◆◆
libnettle																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆◇◇ ◆◆◆
salsa20	◇◇◆ ◇◇◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◆	◆	◇◇◆ ◇◇◇	◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◇	◇	◇
sha512	◇◆◆ ◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇× ◆◇◇	◇◇× ◆◇◇	◆◇× ◆◆◆
hmac	◇◇◆ ◇◇◇	◆	◆	◆	◆	◆	◆	◆	◇◇◆ ◇◇◇	◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◇	◇	◇
ed25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×
x25519	◇×× ×◆◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇×× ×◆◇	◇×× ×◆◇	◇×	◇×
rust-crypto																		
salsa20	◇	◆	◆	◇◆◆ ◆◇◇	◆	◆	◆	◆	◇×	◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◇	◇×	◇
poly1305	◇◇× ◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇× ◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◇× ◆◇◇	◇◇× ◆◇◇	◇×
sha512	◇◇◆ ◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◇ ◆◇◇	◇◇◇ ◆◇◇	◇
x25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
stream-xor	◇◇◇ ◆◇◇	◆	◆	◇◆◇ ◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆ ◆◇◆	◇◆◆ ◆◇◇	◆	◆	◇◇◇ ◆◇◇	◇	◇
libjade																		
salsa20	◇◇◆ ◆◇◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◆	◆	◇◇◆ ◇◇◇	◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◇◇◆ ◆◇◇	◇◇◆ ◇◇◇	◇◇◆ ◇◇◇
poly1305	◇◇◆ ◇◇◇	◆	◇◆◆ ◇◇◇	◇◆◆ ◇◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◇	◆
sha512	◇◇◆ ◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◇◇◆ ◆◇◇	◇◇◆ ◆◇◇	◇◇◆ ◇◇◇
x25519	◇×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇×	◇×	◇×	◇×
stream-xor	◇◇◆ ◆◇◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆ ◆◇◇	◇◆◆ ◆◇◇	◆	◆	◇◇◆ ◆◇◇	◇◇◆ ◇◇◇	◇

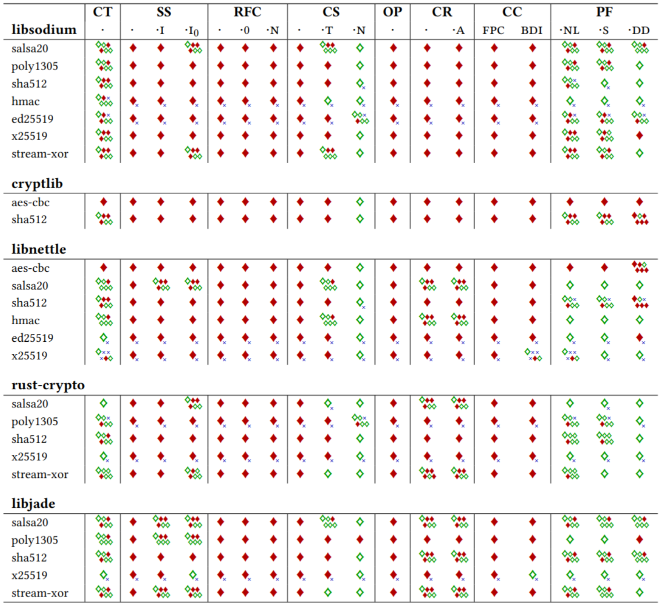
Evaluation results

- All analyzed implementations leak
- Memory-safe languages (e.g., Rust) not significantly mitigate leaks
- CT programming (e.g., libsodium, libjade) does not fully prevent leaks

	CT	SS			RFC			CS			OP	CR		CC		PF		
libsodium	·	·	·I	·I$_0$	·	·0	·N	·	·T	·N	·	·	·A	FPC	BDI	·NL	·S	·DD
salsa20	◇◇◆/◆◇◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◆◆/◇◇◇	◇	◆	◆	◆	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇◇◆/◇◇◇
poly1305	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇×/◆◇◇	◇×	◇
hmac	◇◆×/◇◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
ed25519	◇◆×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◆×/◆◇◇	◇◆×/◆◇◇	◇◇×/◆◇◇
x25519	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◆◇/◆◇◇	◆
stream-xor	◇◆◆/◆◇◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◆◆/◇◇◇	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◇◆/◆◇◇	◇
cryptlib																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆◇◆/◆◆◆
libnettle																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆◆◇/◆◆◆
salsa20	◇◇◆/◇◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇	◇
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇×/◆◇◇	◇◇×/◆◇◇	◆◇×/◆◆◆
hmac	◇◇◆/◇◇◇	◆	◆	◆	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇	◇
ed25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×
x25519	◇××/×◆◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇××/×◆◇	◇××/×◆◇	◇×	◇×
rust-crypto																		
salsa20	◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇×	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇×	◇
poly1305	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◇◇×/◆◇◇	◇×
sha512	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◇/◆◇◇	◇◇◇/◆◇◇	◇
x25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
stream-xor	◇◇◇/◆◇◇	◆	◆	◇◆◇/◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆/◆◇◆	◇◆◆/◆◇◇	◆	◆	◇◇◇/◆◇◇	◇	◇
libjade																		
salsa20	◇◇◆/◆◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◇◇◇	◇◇◆/◇◇◇
poly1305	◇◇◆/◇◇◇	◆	◇◆◆/◇◇◇	◇◆◆/◇◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◇	◆
sha512	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇◇◆/◇◇◇
x25519	◇×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇×	◇×	◇×	◇×
stream-xor	◇◇◆/◆◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◇◇◇	◇

Evaluation results

- All analyzed implementations leak
- Memory-safe languages (e.g., Rust) not significantly mitigate leaks
- CT programming (e.g., libsodium, libjade) does not fully prevent leaks
- Majority of leaks are present even without prediction clause



	CT	SS			RFC			CS			OP	CR		CC		PF		
libsodium	·	·	·I	·I_0	·	·0	·N	·	·T	·N	·	·	·A	FPC	BDI	·NL	·S	·DD
salsa20	◇◇◆/◆◇◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◆◆/◇◇◇	◇	◆	◆	◆	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇◇◆/◇◇◇
poly1305	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇×/◆◇◇	◇×	◇
hmac	◇◆×/◇◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
ed25519	◇◆×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◆×/◆◇◇	◇◆×/◆◇◇	◇◇×/◆◇◇
x25519	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◆◇/◆◇◇	◆
stream-xor	◇◆◆/◆◇◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◆◆/◇◇◇	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◇◆/◆◇◇	◇
cryptlib																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆◇◆/◆◆◆
libnettle																		
aes-cbc	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◆	◆	◆◆◇/◆◆◆
salsa20	◇◇◆/◇◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇	◇
sha512	◇◆◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇×	◆	◆	◆	◆	◆	◇◇×/◆◇◇	◇◇×/◆◇◇	◆◇×/◆◆◆
hmac	◇◇◆/◇◇◇	◆	◆	◆	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇	◇
ed25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◆×
x25519	◇××/×◆◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇××/×◆◇	◇××/×◆◇	◇×	◇×
rust-crypto																		
salsa20	◇	◆	◆	◇◆◆/◆◇◇	◆	◆	◆	◆	◇×	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇	◇×	◇
poly1305	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◆×	◆×	◆×	◆×	◆×	◇◇×/◆◇◇	◇◇×/◆◇◇	◇×
sha512	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◆	◆	◆	◆	◇◇◇/◆◇◇	◇◇◇/◆◇◇	◇
x25519	◇×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◇×	◇×
stream-xor	◇◇◇/◆◇◇	◆	◆	◇◆◇/◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆/◆◇◆	◇◆◆/◆◇◇	◆	◆	◇◇◇/◆◇◇	◇	◇
libjade																		
salsa20	◇◇◆/◆◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇◇◆/◇◇◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◇◇◇	◇◇◆/◇◇◇
poly1305	◇◇◆/◇◇◇	◆	◇◆◆/◇◇◇	◇◆◆/◇◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◇	◇	◆
sha512	◇◇◆/◆◇◇	◆	◆	◆	◆	◆	◆	◆	◆	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◆◇◇	◇◇◆/◇◇◇
x25519	◇×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◆×	◇×	◆×	◆×	◆×	◆×	◇×	◇×	◇×	◇×
stream-xor	◇◇◆/◆◇◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◆	◆	◇	◇	◆	◇◆◆/◆◇◇	◇◆◆/◆◇◇	◆	◆	◇◇◆/◆◇◇	◇◇◆/◇◇◇	◇

Evaluation results

- All analyzed implementations leak
- Memory-safe languages (e.g., Rust) not significantly mitigate leaks
- CT programming (e.g., libsodium, libjade) does not fully prevent leaks
- Majority of leaks are present even without prediction clause

$\langle\textsc{Seq}\rangle\ \langle\textsc{Pht}\rangle\ \langle\textsc{Sls}\rangle$
$\langle\textsc{Stl}\rangle\ \langle\textsc{Rsb}_\perp\rangle\ \langle\textsc{Rsb}_\circ\rangle$

◇◇×
♦◇◇

	CT	SS			RFC			CS			OP	CR		CC		PF		
libsodium	·	·	·I	·I$_0$	·	·0	·N	·	·T	·N	·	·	·A	FPC	BDI	·NL	·S	·DD
salsa20	◇◇♦/♦◇◇	♦	♦	◇♦♦/♦◇◇	♦	♦	♦	♦	◇♦♦/◇◇◇	◇	♦	♦	♦	♦	♦	◇◇♦/♦◇◇	◇◇♦/♦◇◇	◇◇♦/◇◇◇
poly1305	◇◇♦/♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇◇♦/♦◇◇	◇◇♦/♦◇◇	◇
sha512	◇♦♦/♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇×	♦	♦	♦	♦	♦	◇◇×/♦◇◇	◇×	◇
hmac	◇♦×/◇◇◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	◇×
ed25519	◇♦×/♦◇◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇◇×/♦◇◇	♦×	♦×	♦×	♦×	♦×	◇♦×/♦◇◇	◇♦×/♦◇◇	◇◇×/♦◇◇
x25519	◇♦♦/♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇♦♦/♦◇◇	◇♦◇/♦◇◇	♦
stream-xor	◇♦♦/♦◇◇	♦	♦	◇♦♦/♦◇◇	♦	♦	♦	♦	◇♦♦/◇◇◇	◇	♦	♦	♦	♦	♦	◇♦♦/♦◇◇	◇◇♦/♦◇◇	◇
cryptlib																		
aes-cbc	♦	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	♦	♦	♦
sha512	◇♦♦/♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦◇♦/♦♦♦
libnettle																		
aes-cbc	♦	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	♦	♦	♦♦◇/♦♦♦
salsa20	◇◇♦/◇◇◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	♦	♦	◇◇♦/◇◇◇	◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	◇	◇	◇
sha512	◇♦♦/♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇×	♦	♦	♦	♦	♦	◇◇×/♦◇◇	◇◇×/♦◇◇	♦◇×/♦♦♦
hmac	◇◇♦/◇◇◇	♦	♦	♦	♦	♦	♦	♦	◇◇♦/◇◇◇	◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	◇	◇	◇
ed25519	◇×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	♦×
x25519	◇××/×♦◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	◇××/×♦◇	◇××/×♦◇	◇×	◇×
rust-crypto																		
salsa20	◇	♦	♦	◇♦♦/♦◇◇	♦	♦	♦	♦	◇×	◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	◇	◇×	◇
poly1305	◇◇×/♦◇◇	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇◇×/♦◇◇	♦×	♦×	♦×	♦×	♦×	◇◇×/♦◇◇	◇◇×/♦◇◇	◇×
sha512	◇◇♦/♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	♦	♦	♦	♦	◇◇◇/♦◇◇	◇◇◇/♦◇◇	◇
x25519	◇×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	◇×	◇×
stream-xor	◇◇◇/♦◇◇	♦	♦	◇♦◇/♦◇◇	♦	♦	♦	♦	◇	◇	♦	◇♦♦/♦◇♦	◇♦♦/♦◇◇	♦	♦	◇◇◇/♦◇◇	◇	◇
libjade																		
salsa20	◇◇♦/♦◇◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	♦	♦	◇◇♦/◇◇◇	◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	◇◇♦/♦◇◇	◇◇♦/◇◇◇	◇◇♦/◇◇◇
poly1305	◇◇♦/◇◇◇	♦	◇♦♦/◇◇◇	◇♦♦/◇◇◇	♦	♦	♦	♦	♦	♦	♦	♦	♦	♦	♦	◇	◇	♦
sha512	◇◇♦/♦◇◇	♦	♦	♦	♦	♦	♦	♦	♦	◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	◇◇♦/♦◇◇	◇◇♦/♦◇◇	◇◇♦/◇◇◇
x25519	◇×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	♦×	◇×	♦×	♦×	♦×	♦×	◇×	◇×	◇×	◇×
stream-xor	◇◇♦/♦◇◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	♦	♦	◇	◇	♦	◇♦♦/♦◇◇	◇♦♦/♦◇◇	♦	♦	◇◇♦/♦◇◇	◇◇♦/◇◇◇	◇

Example: leak from constant-time swap

```
fe25519_cswap(fe25519_limb f[5], fe25519_limb g[5], bool b)
{
    mask = (-(int64_t) b);

    x[0..5] = f[0..5] ^ g[0..5];

    x[0..5] &= mask;

    f[0..5] = f[0..5] ^ x[0..5];
    g[0..5] = f[0..5] ^ x[0..5];
}
```

Example: leak from constant-time swap

secret

```
fe25519_cswap(fe25519_limb f[5], fe25519_limb g[5], bool b)
{
    mask = (-(int64_t) b);

    x[0..5] = f[0..5] ^ g[0..5];

    x[0..5] &= mask;

    f[0..5] = f[0..5] ^ x[0..5];
    g[0..5] = f[0..5] ^ x[0..5];
}
```

Example: leak from constant-time swap

secret

```
fe25519_cswap(fe25519_limb f[5], fe25519_limb g[5], bool b)
{
    mask = (-(int64_t) b);

    x[0..5] = f[0..5] ^ g[0..5];

    x[0..5] &= mask;

    f[0..5] = f[0..5] ^ x[0..5];
    g[0..5] = f[0..5] ^ x[0..5];
}
```

mask $= 0 \rightarrow$ result $= 0$
RFC: compress to zero register

Example: leak from constant-time swap

secret

```
fe25519_cswap(fe25519_limb f[5], fe25519_limb g[5], bool b)
{
    mask = (-(int64_t) b);

    x[0..5] = f[0..5] ^ g[0..5];

    x[0..5] &= mask;

    f[0..5] = f[0..5] ^ x[0..5];
    g[0..5] = f[0..5] ^ x[0..5];
}
```

mask $= 0 \rightarrow$ result $= 0$
RFC: compress to zero register

mask $= 0 \rightarrow$ x $= 0 \rightarrow$ xor 0 not changed
CS: xor is simplified

Example: leak from constant-time swap

secret

```
fe25519_cswap(fe25519_limb f[5], fe25519_limb g[5], bool b)
{
    mask = (-(int64_t) b);

    x[0..5] = f[0..5] ^ g[0..5];

    x[0..5] &= mask;

    f[0..5] = f[0..5] ^ x[0..5];
    g[0..5] = f[0..5] ^ x[0..5];
}
```

`mask` $= 0 \rightarrow$ result $= 0$
RFC: compress to zero register

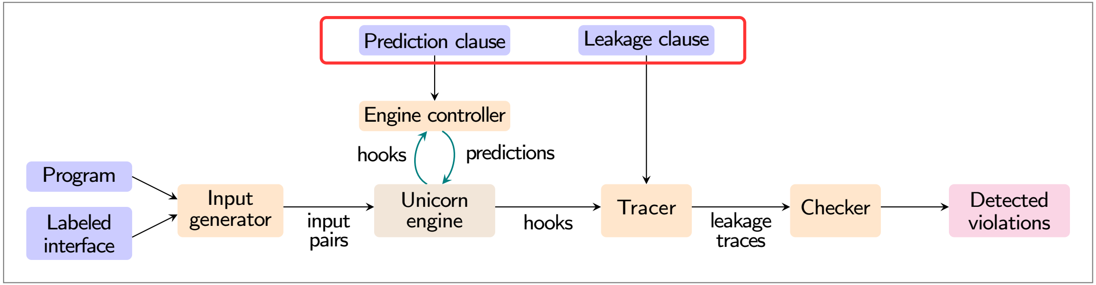
`mask` $= 0 \rightarrow$ `x` $= 0 \rightarrow$ `xor` 0 not changed
CS: `xor` is simplified

values not swapped $\rightarrow$ memory not modified
SS: stores are suppressed

Summary

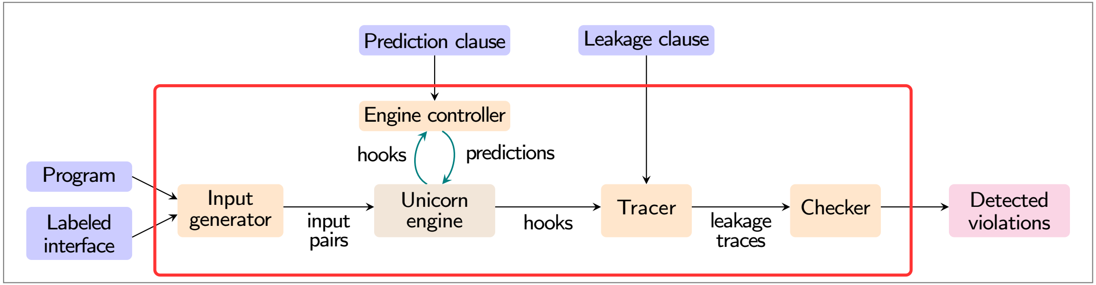
Summary

- LmSpec: Language for specifying leakage models at the ISA level



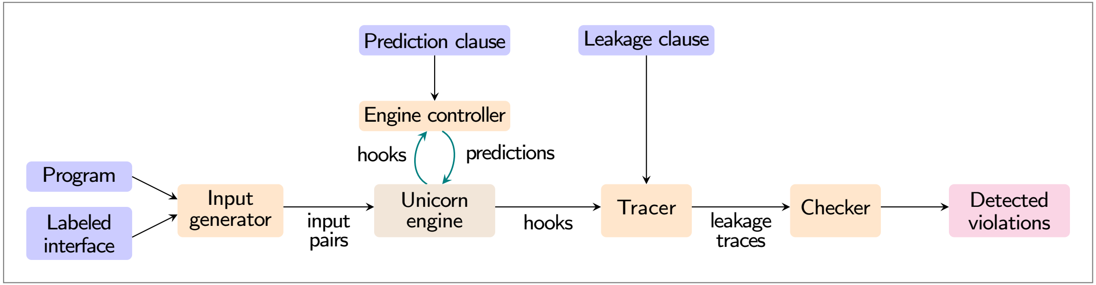
Summary

- LmSpec: Language for specifying leakage models at the ISA level
- LmTest: Testing framework for automatically detecting leaks



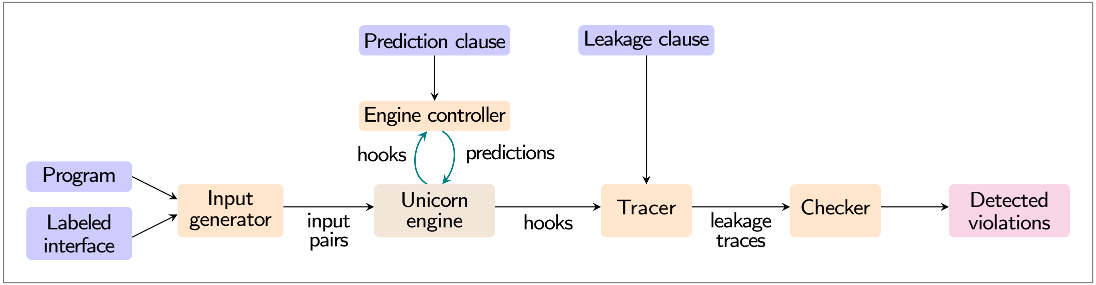
Summary

- LmSpec: Language for specifying leakage models at the ISA level
- LmTest: Testing framework for automatically detecting leaks
- Extensive evaluation on cryptographic algorithms → *all leak!* 😱



Summary

- LmSpec: Language for specifying leakage models at the ISA level
- LmTest: Testing framework for automatically detecting leaks
- Extensive evaluation on cryptographic algorithms → *all leak!* 😱
- For more details, see `https://arxiv.org/pdf/2402.00641`


Prediction clause
Leakage clause
Engine controller
hooks
predictions
Program
Labeled interface
Input generator
input pairs
Unicorn engine
hooks
Tracer
leakage traces
Checker
Detected violations