SoK: Automated Kernel Vulnerability Discovery and Exploit Generation

Anil Kurmus, **Andrea Mambretti**, Alessandro Sorniotti

IBM Research Europe – Zurich

Vincent Lenders, Damian Pfammatter, Bernhard Tellenbach

Armasuisse - Cyber Defense Campus

WOOT 2025

armasuisse

IBM

Once upon a time...

armasuisse

IBM

Once upon a time...

UNIVERSITY OF OXFORD
COMPUTING LABORATORY

MSc COMPUTER SCIENCE DISSERTATION

Automatic Generation of Control Flow Hijacking Exploits for Software Vulnerabilities

---

Author:
Sean HEELAN

Supervisor:
Dr. Daniel KROENING

2009

armasuisse

IBM

Once upon a time...

2011

AEG: Automatic Exploit Generation

Thanassis Avgerinos, Sang Kil Cha, Brent Lim Tze Hao and David Brumley
Carnegie Mellon University, Pittsburgh, PA
{thanassis, sangkilc, brentlim, dbrumley}@cmu.edu

armasuisse

IBM

Once upon a time…

2016

armasuisse

IBM

Darpa - Cyber Grand Challenge

armasuisse

IBM

Darpa - Cyber Grand Challenge

Contributions:

First big push towards full end-2-end AEG

Considerably advanced static and dynamic analyses

Modern opensource tools (e.g., angr, S2E)

Darpa - Cyber Grand Challenge

Contributions:

First big push towards full end-2-end AEG

Considerably advanced static and dynamic analyses

Modern opensource tools (e.g., angr, S2E)

Limitations:

Simplified OS & Architecture (e.g., 7 syscalls)

Small size software

Binary code only

Proof-of-vulnerability only (e.g., IP control)

Darpa - Cyber Grand Challenge

Contributions:

First big push towards full end-2-end AEG

Considerably advanced static and dynamic analyses

Modern opensource tools (e.g., angr, S2E)

Limitations:

Simplified OS & Architecture (e.g., 7 syscalls)

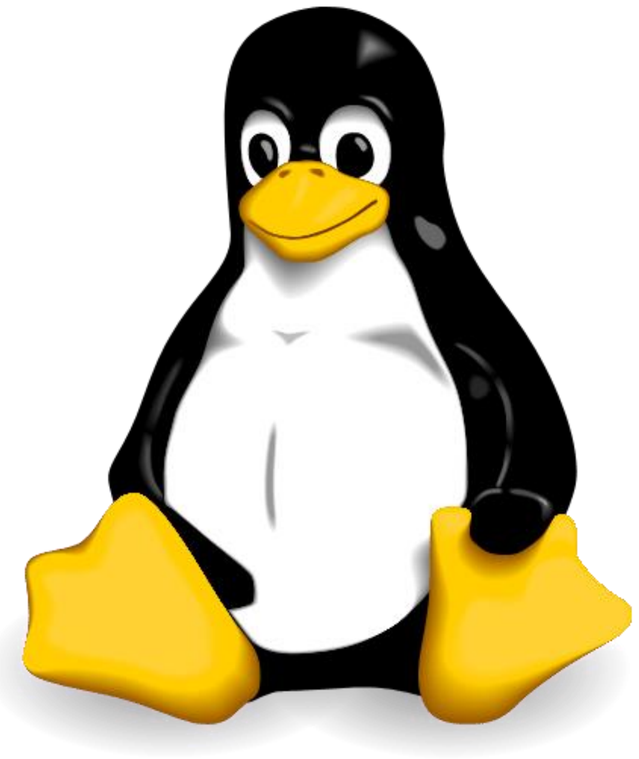
Small size software

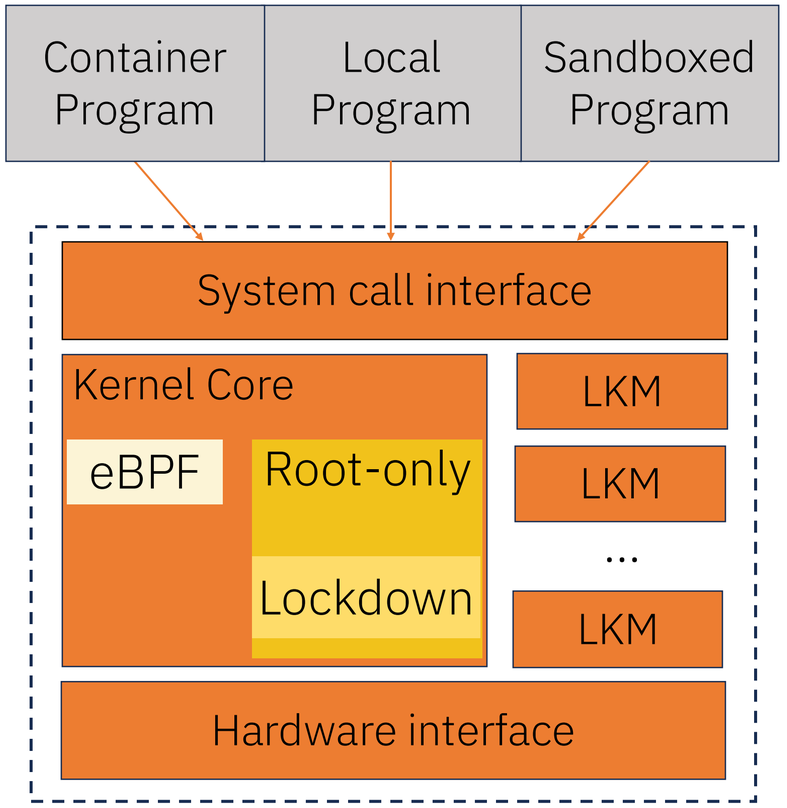
Binary code only

Proof-of-vulnerability only (e.g., IP control)

~10 years to move it to the next level

The ultimate target





Attacker

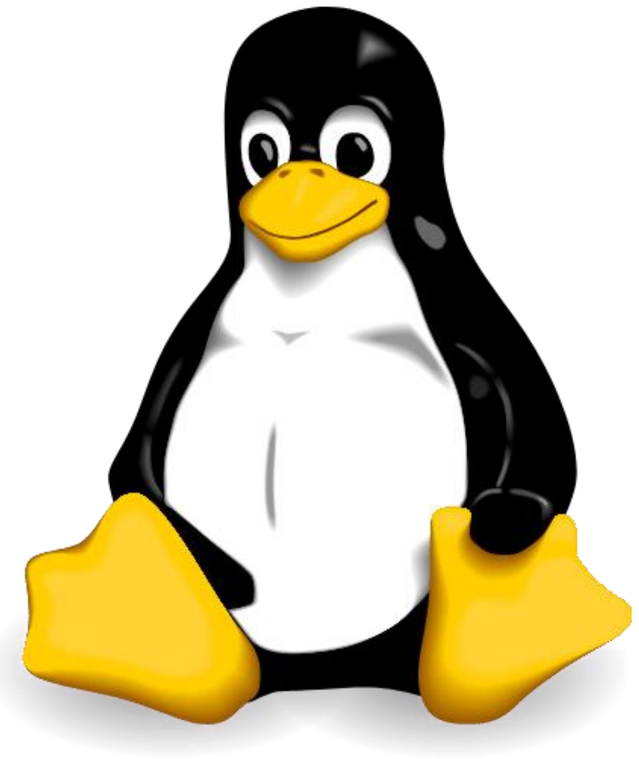
Attack surface

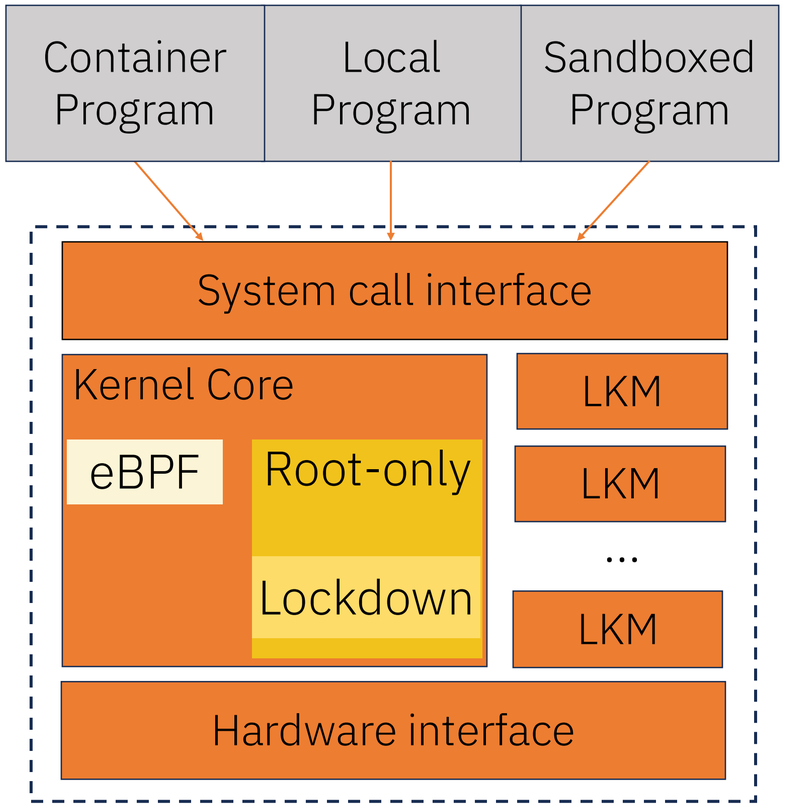
Running Kernel

armasuisse

IBM

The ultimate target





Attacker

Attack surface

------- Running Kernel

High Privilege

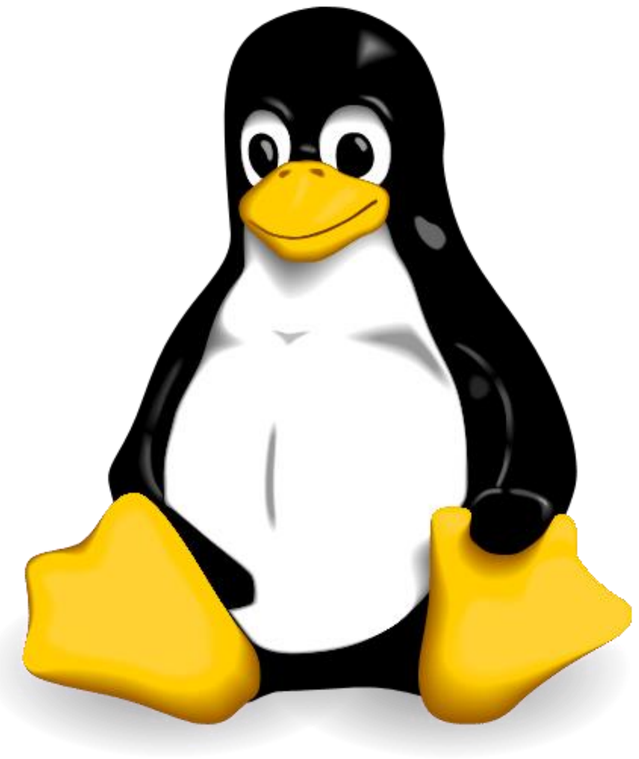
Large attack surface

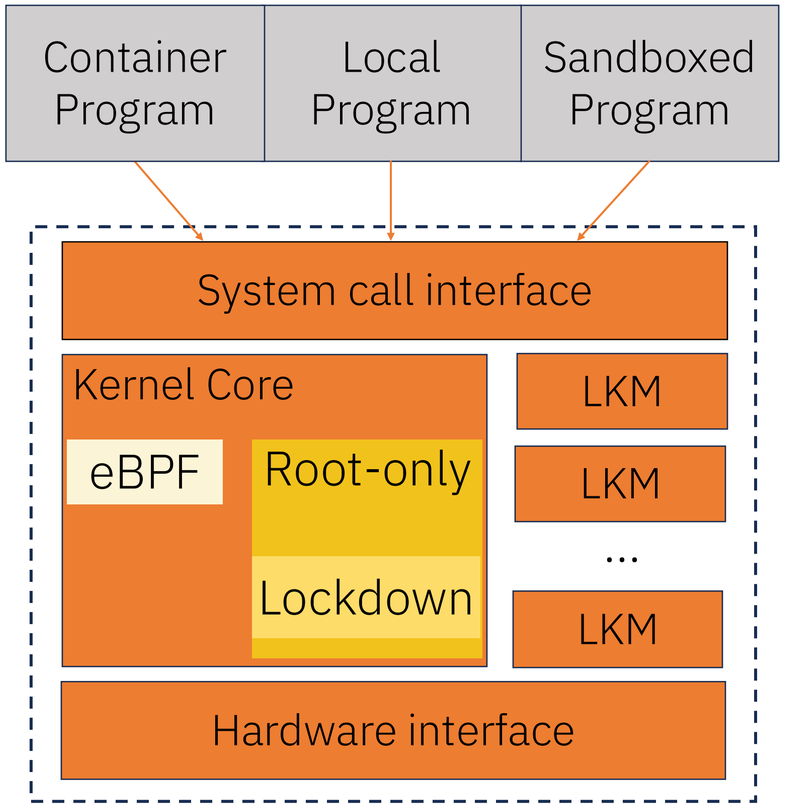
Widely available and used

armasuisse

IBM

The ultimate target





Attacker

Attack surface

Running Kernel

High Privilege

Large attack surface

Widely available and used

40 Millions LoC

Several threat models

Several attacks/mitigations

armasuisse

IBM

Threat models

Local

Local/User sandbox

Local/Container

Remote

Kernel sandbox (eBPF)

Lockdown

Physical

Vulnerability classes

PRECURSOR: might lead to a serious consequence only through a base vulnerability

HELPER: might only lead to an attacker achieving its goal in combination with another vulnerability

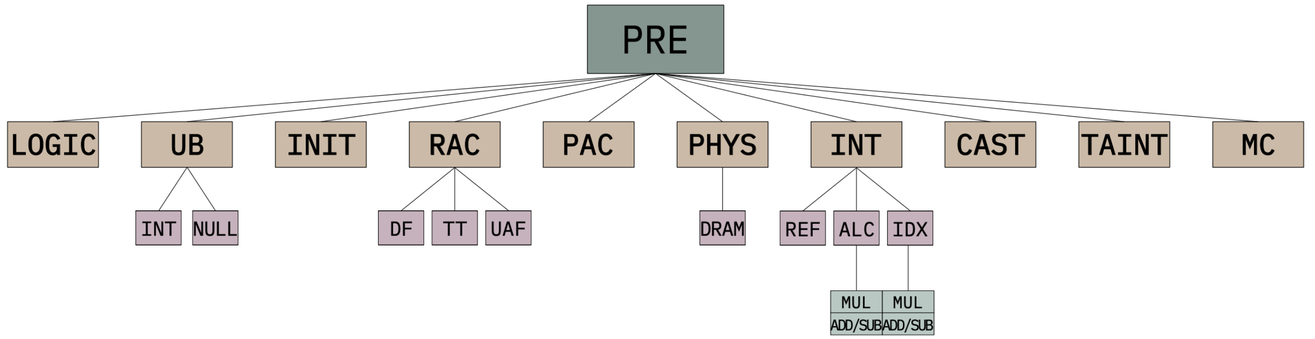
BASE: potentially, it allows attackers to directly reach the goal

Vulnerability classes

PRECURSOR: might lead to a serious consequence only through a base vulnerability

HELPER: might only lead to an attacker achieving its goal in combination with another vulnerability

BASE: potentially, it allows attackers to directly reach the goal



armasuisse

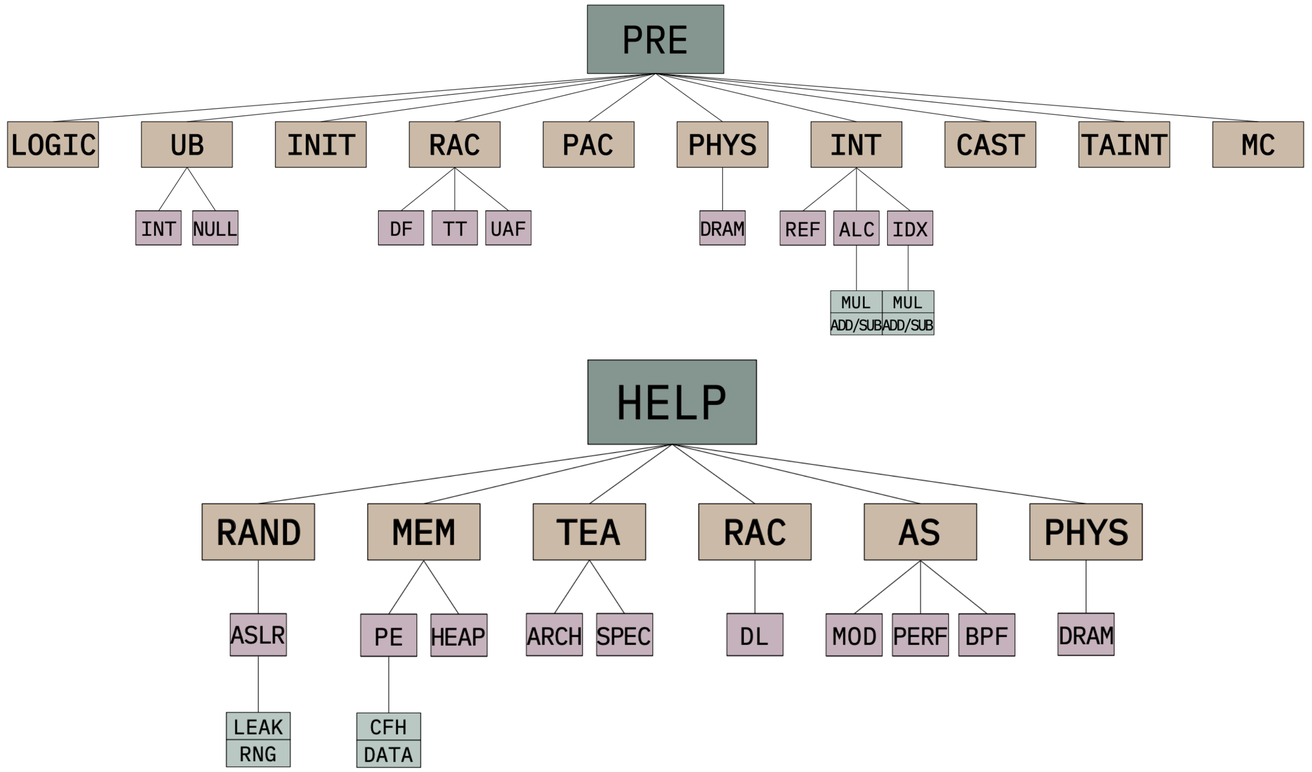
IBM

Vulnerability classes

PRECURSOR: might lead to a serious consequence only through a base vulnerability

HELPER: might only lead to an attacker achieving its goal in combination with another vulnerability

BASE: potentially, it allows attackers to directly reach the goal



armasuisse

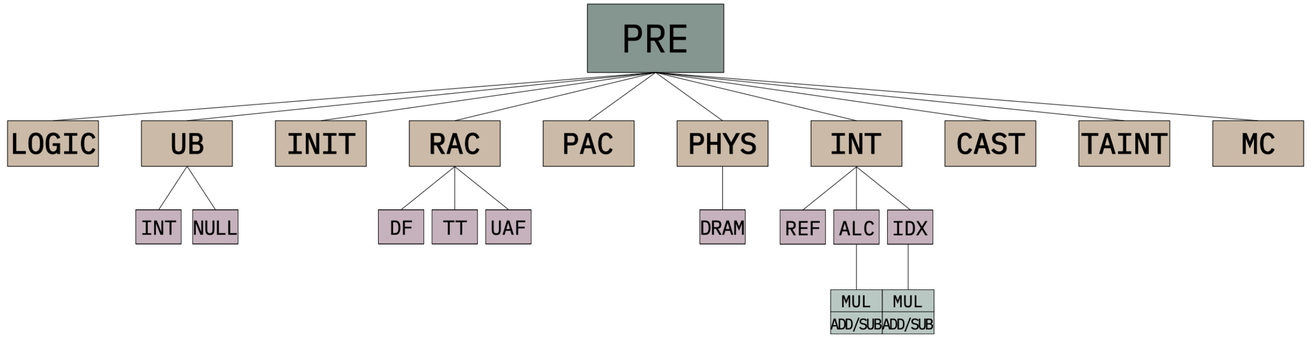
IBM

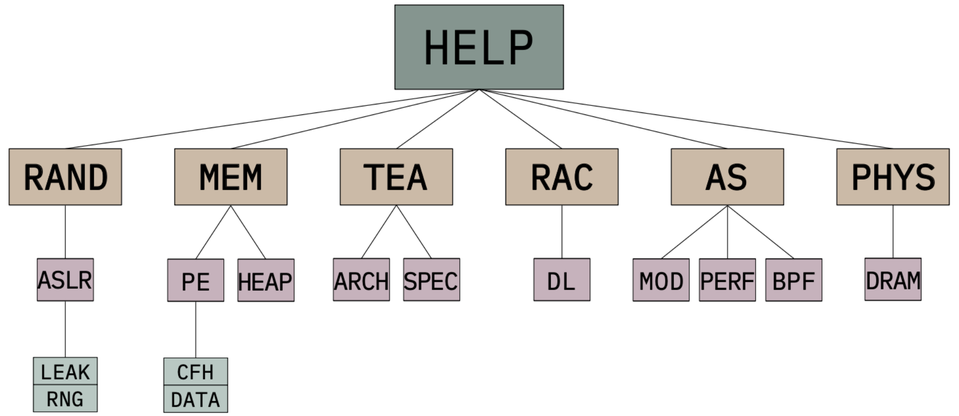
Vulnerability classes

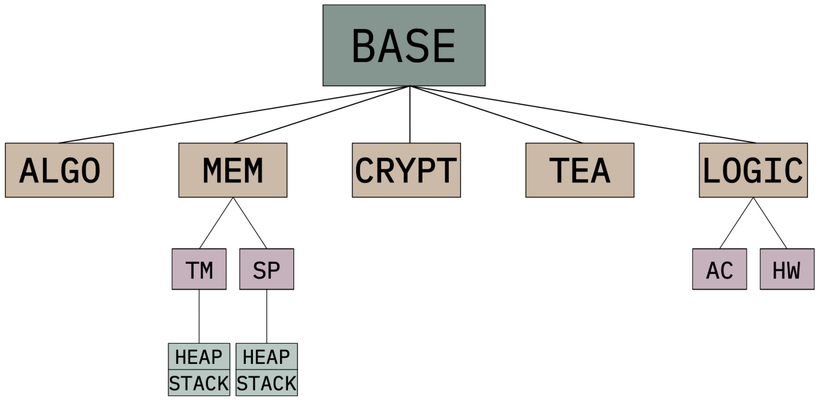
PRECURSOR: might lead to a serious consequence only through a base vulnerability

HELPER: might only lead to an attacker achieving its goal in combination with another vulnerability

BASE: potentially, it allows attackers to directly reach the goal







armasuisse

IBM

Let’s now consider CVE-2021-4083

TREAD 1

```
init_skb()



fd = acquire_fd() {

/* window of 12 insts
between getting fd and
refcnt++ */

}



recvmsg(fd)
```

USD GC

```
if (refcnt == my_refcnt){

  /* I own skb */

...

  delete(skb)

  /* without locking */

}
```

armasuisse

https://project-zero.issues.chromium.org/issues/42451370

IBM

Let’s now consider CVE-2021-4083

TREAD 1

```
init_skb()


fd = acquire_fd() {

/* window of 12 insts
between getting fd and
refcnt++ */

}



recvmsg(fd)
```

USD GC

```
if (refcnt == my_refcnt){

  /* I own skb */

...

  delete(skb)

  /* without locking */

}
```

armasuisse

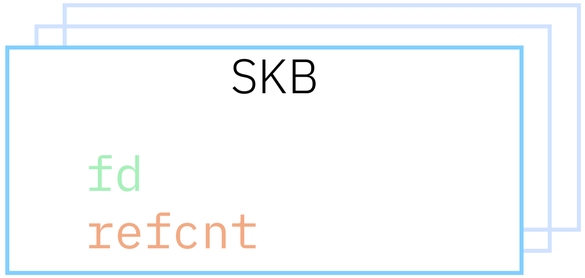
https://project-zero.issues.chromium.org/issues/42451370

IBM

Let’s now consider CVE-2021-4083

TREAD 1

```
init_skb()


fd = acquire_fd() {

/* window of 12 insts
between getting fd and
refcnt++ */

}



recvmsg(fd)
```



USD GC

```
if (refcnt == my_refcnt){

  /* I own skb */

...

  delete(skb)

  /* without locking */

}
```

armasuisse

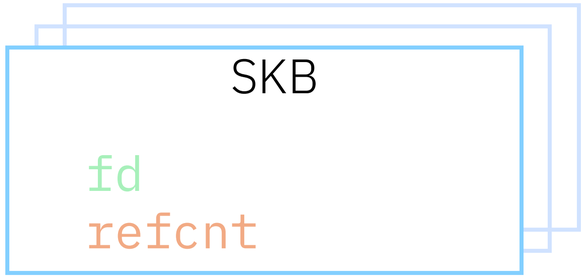
https://project-zero.issues.chromium.org/issues/42451370

IBM

Let’s now consider CVE-2021-4083

TREAD 1

```
init_skb()

fd = acquire_fd() {

/* window of 12 insts
between getting fd and
refcnt++ */

}

recvmsg(fd)
```



USD GC

```
if (refcnt == my_refcnt){

  /* I own skb */

...

  delete(skb)

  /* without locking */

}
```

armasuisse

https://project-zero.issues.chromium.org/issues/42451370

IBM

Let’s now consider CVE-2021-4083

TREAD 1

```
init_skb()

fd = acquire_fd() {

/* window of 12 insts
between getting fd and
refcnt++ */

}

recvmsg(fd)
```

SKB

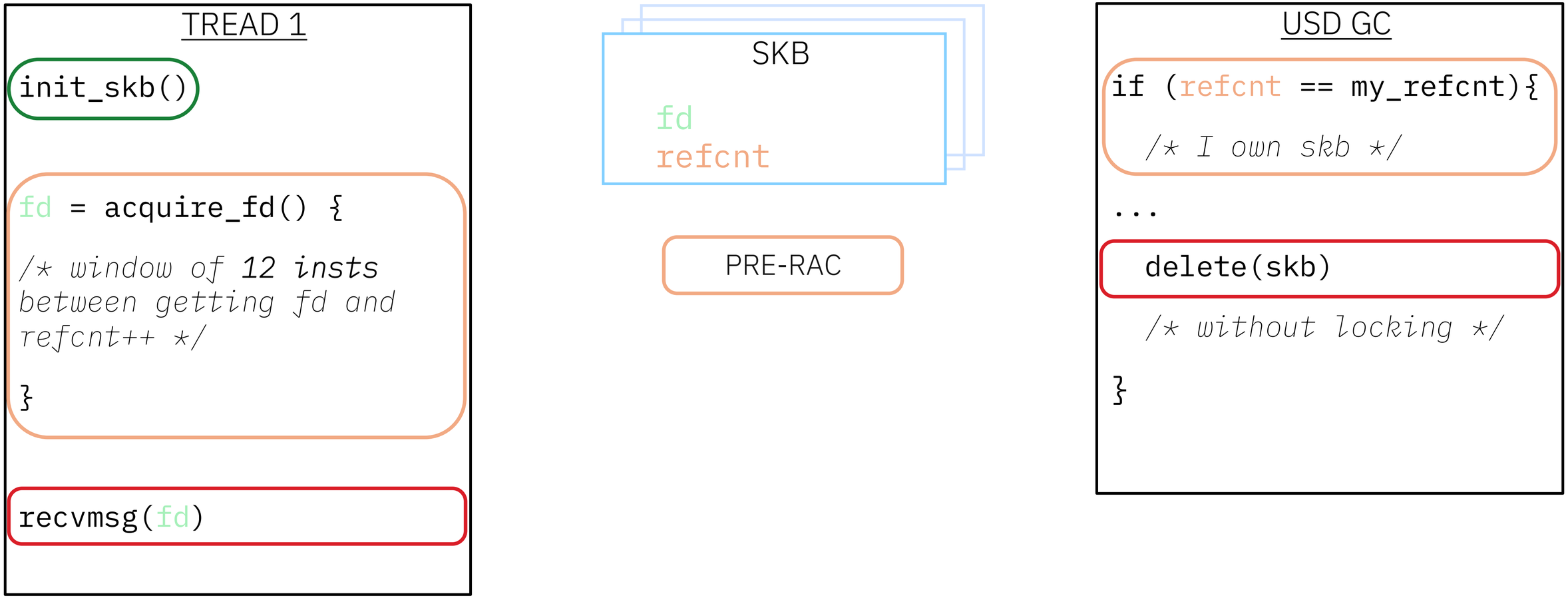
```
fd
refcnt
```

PRE-RAC

USD GC

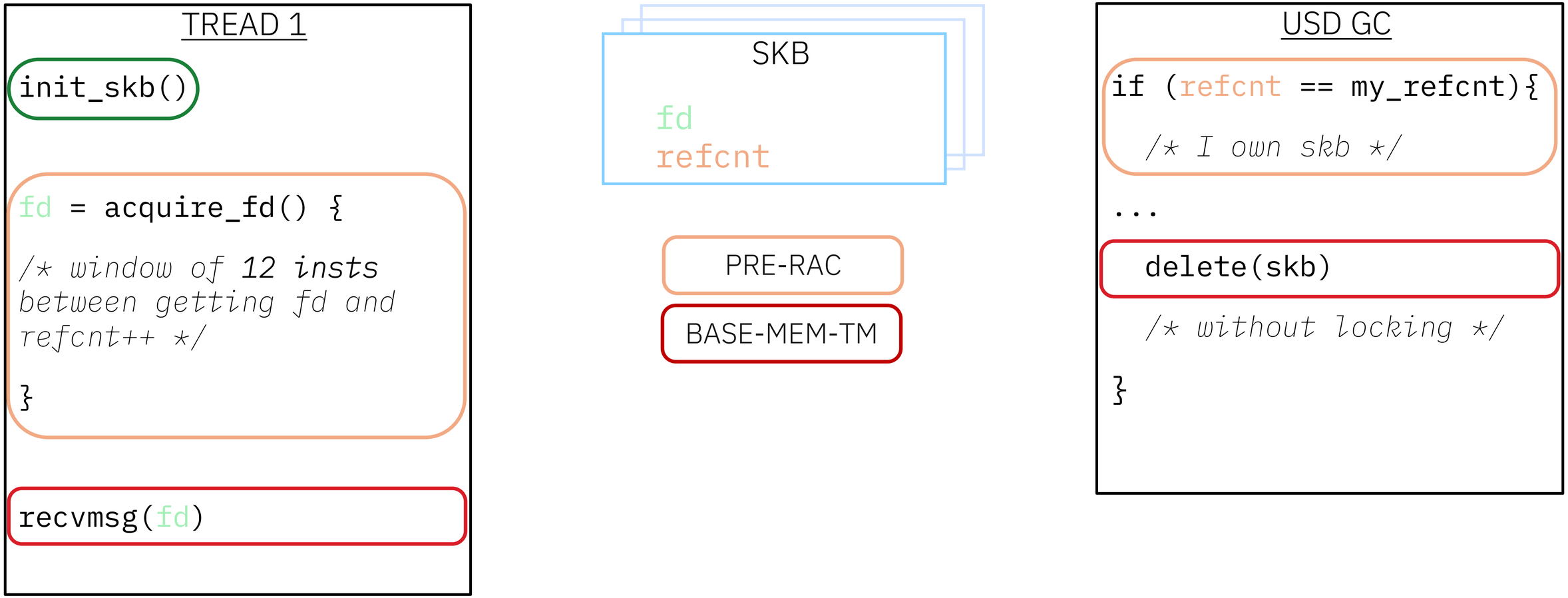
```
if (refcnt == my_refcnt){

  /* I own skb */

...

  delete(skb)

  /* without locking */

}
```

Let’s now consider CVE-2021-4083



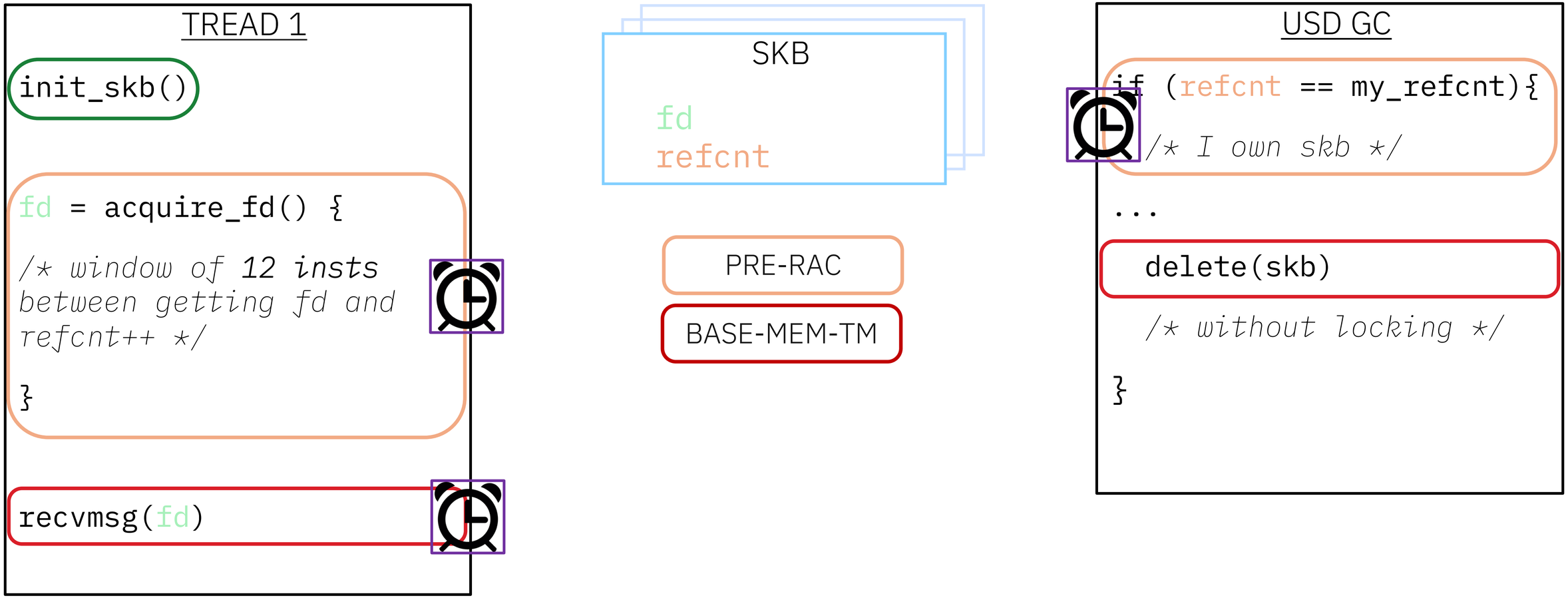
https://project-zero.issues.chromium.org/issues/42451370

Let’s now consider CVE-2021-4083



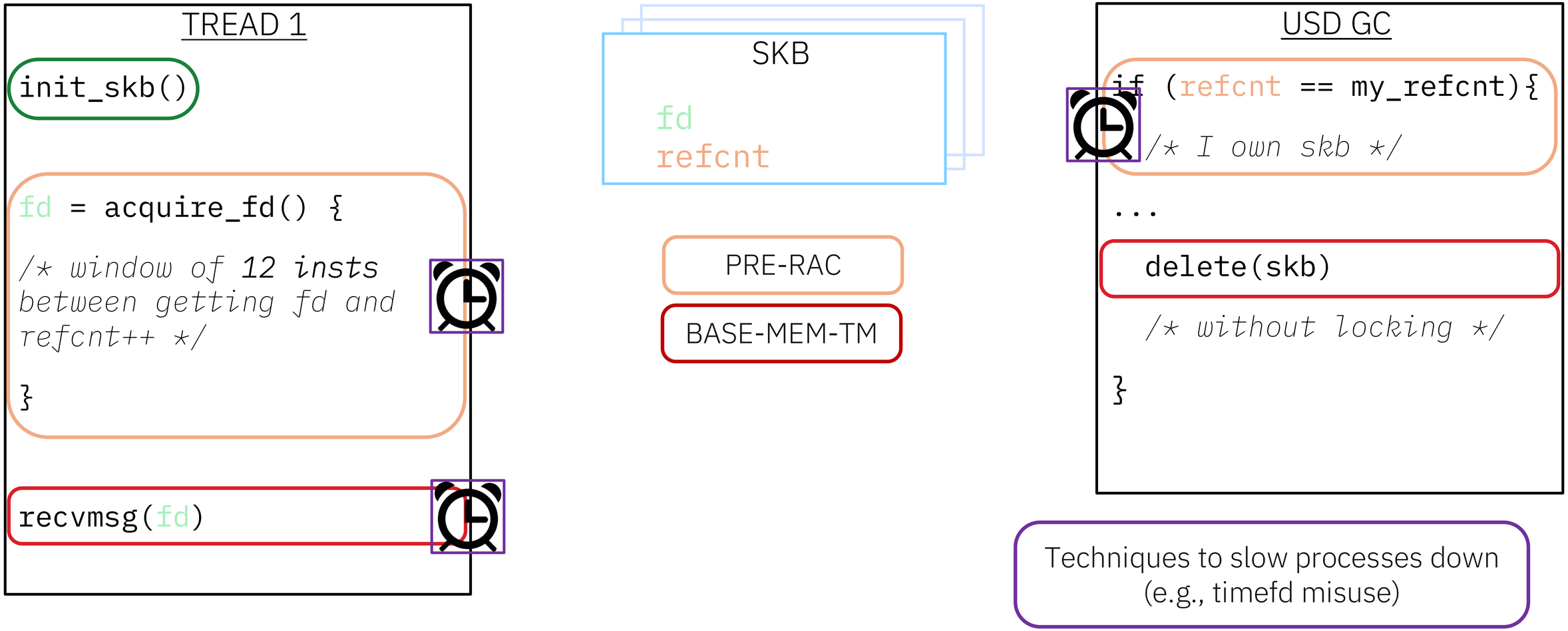
https://project-zero.issues.chromium.org/issues/42451370

Let’s now consider CVE-2021-4083



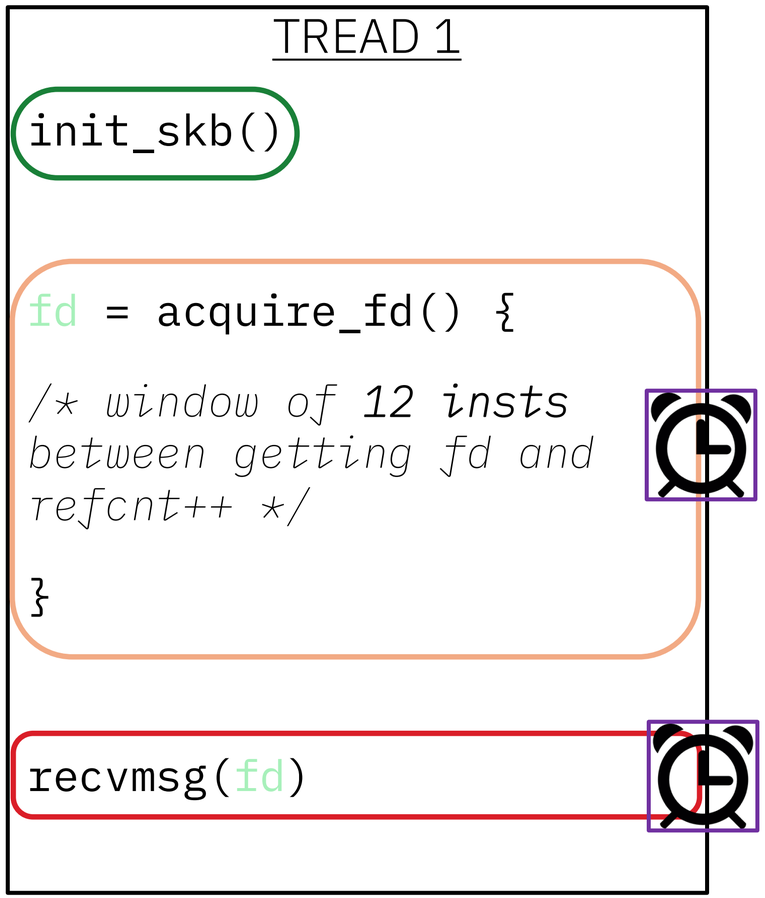
https://project-zero.issues.chromium.org/issues/42451370

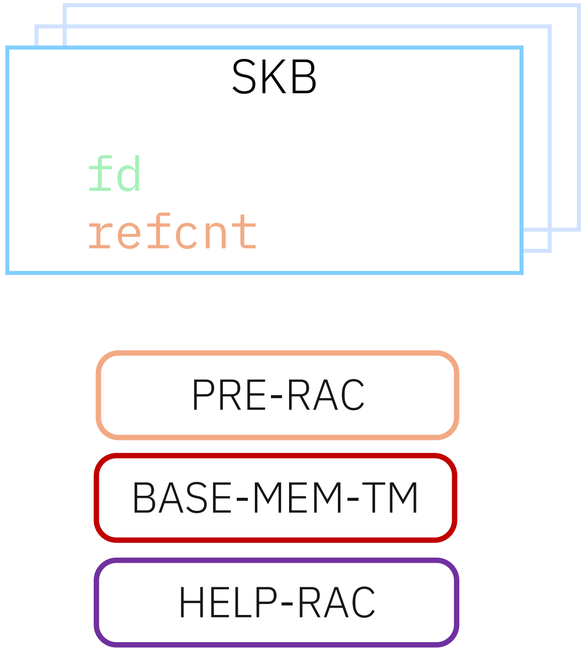
Let’s now consider CVE-2021-4083

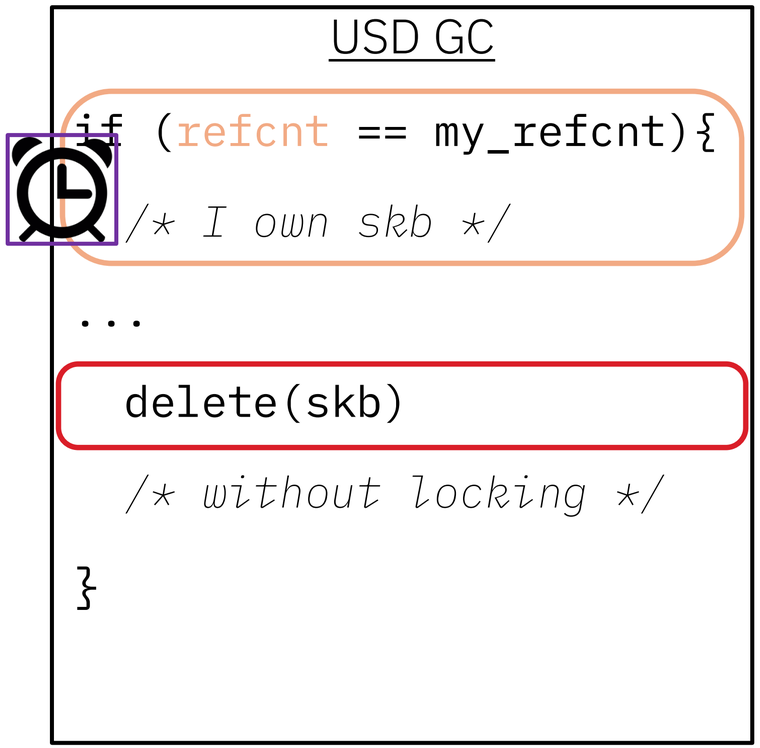


https://project-zero.issues.chromium.org/issues/42451370

Let’s now consider CVE-2021-4083







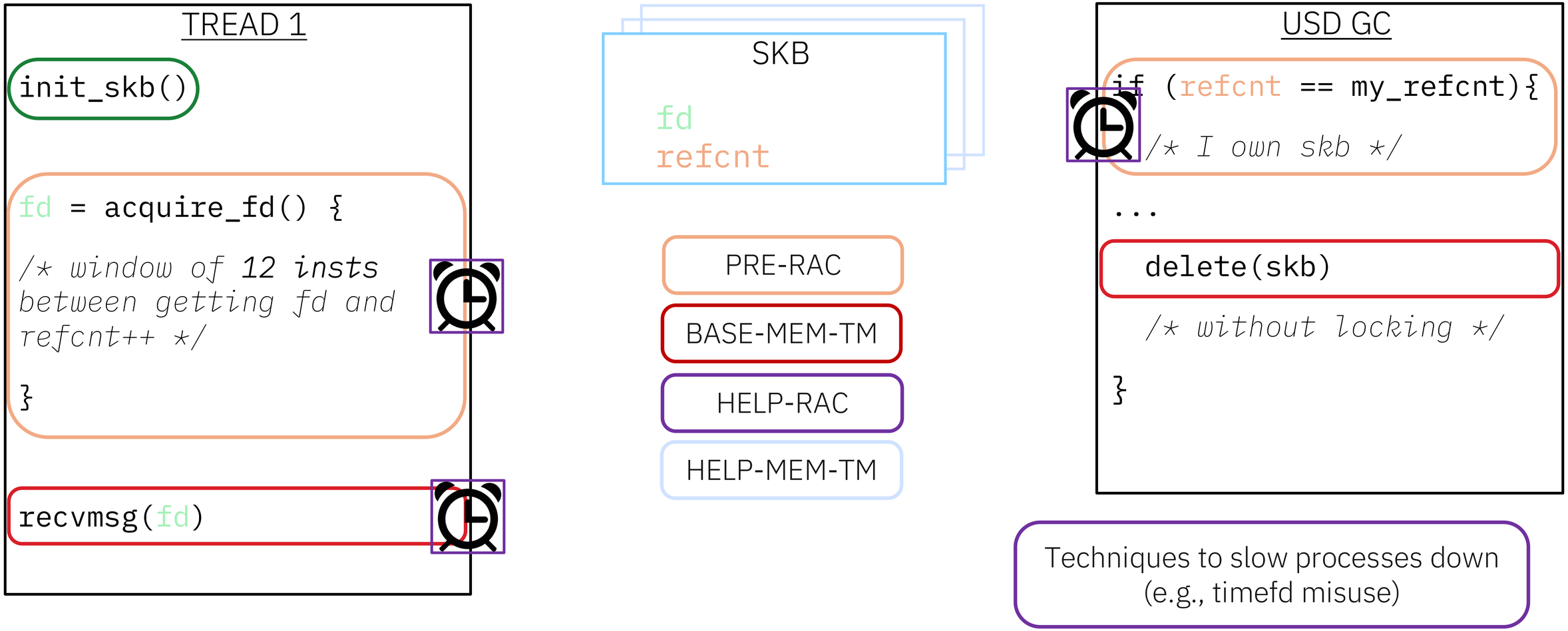
Techniques to slow processes down (e.g., timefd misuse)

armasuisse

https://project-zero.issues.chromium.org/issues/42451370

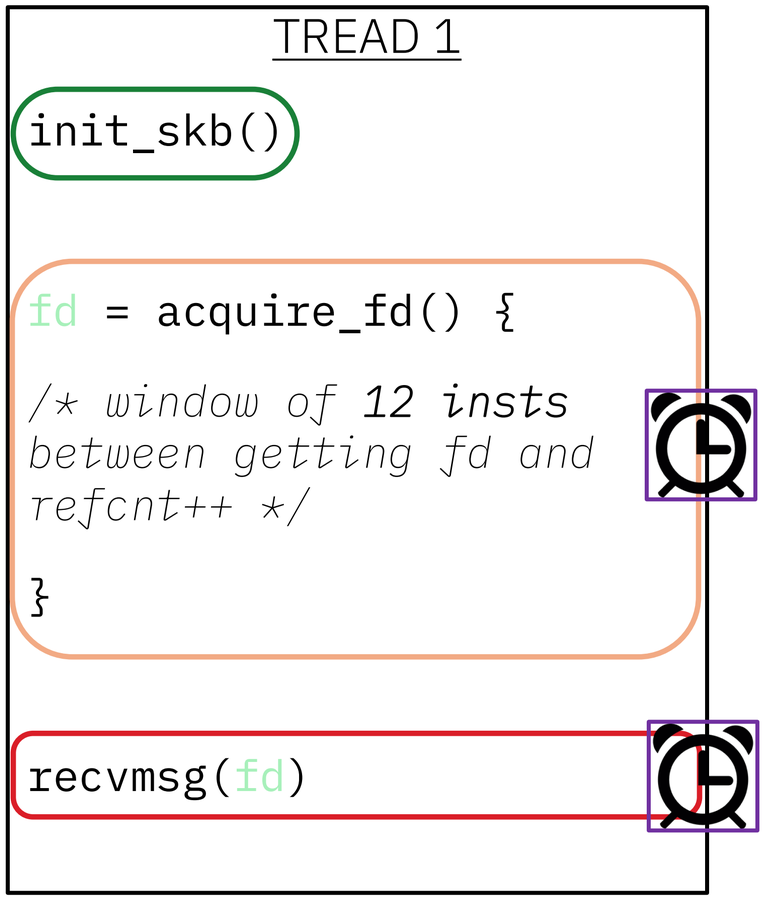
IBM

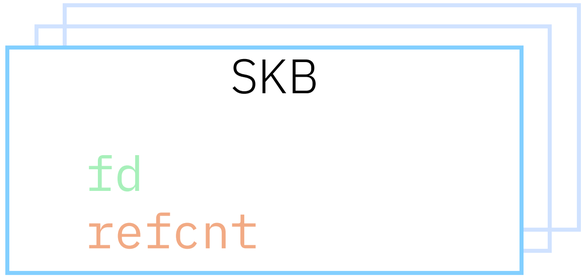
Let’s now consider CVE-2021-4083



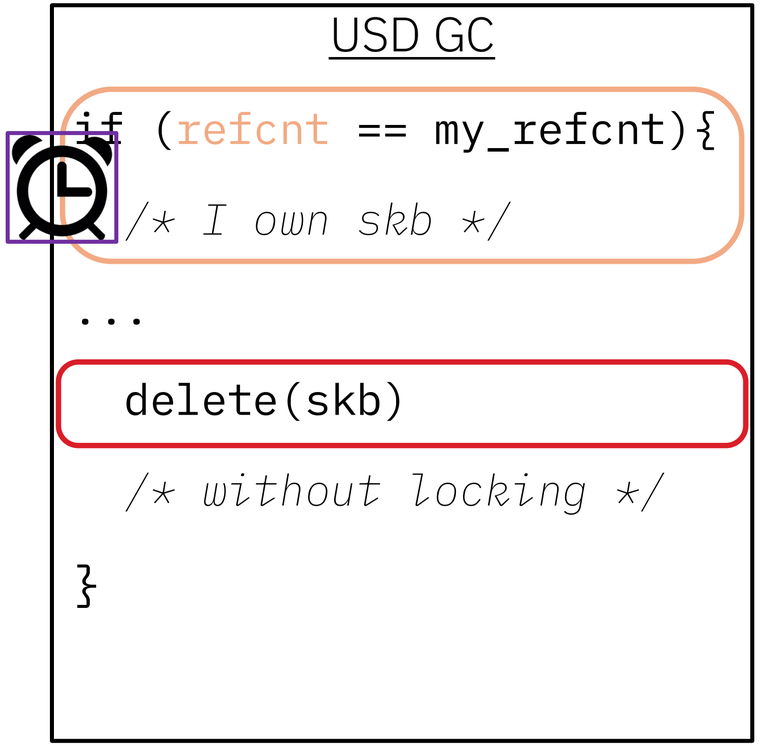
https://project-zero.issues.chromium.org/issues/42451370

Let’s now consider CVE-2021-4083





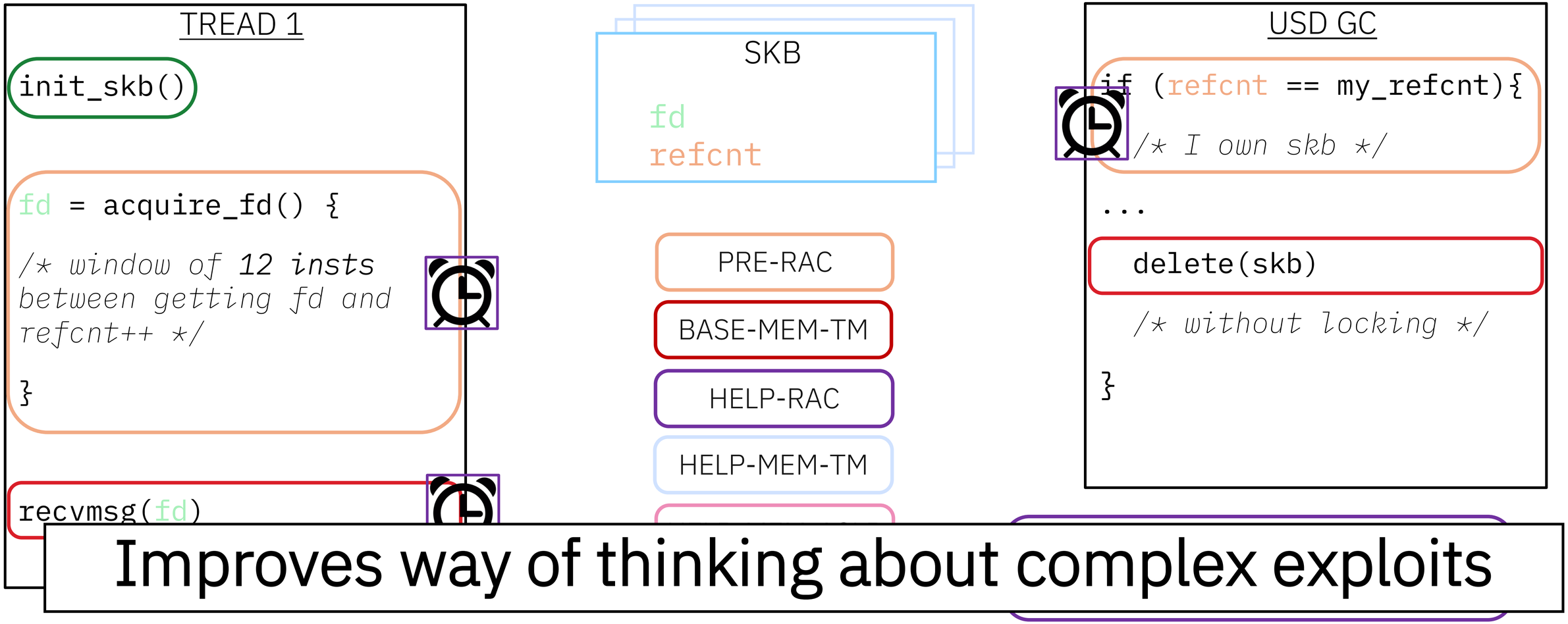
- PRE-RAC
- BASE-MEM-TM
- HELP-RAC
- HELP-MEM-TM
- HELP-MEM-ASLR



Techniques to slow processes down (e.g., timefd misuse)

https://project-zero.issues.chromium.org/issues/42451370

Let’s now consider CVE-2021-4083



Improves way of thinking about complex exploits

armasuisse

https://project-zero.issues.chromium.org/issues/42451370

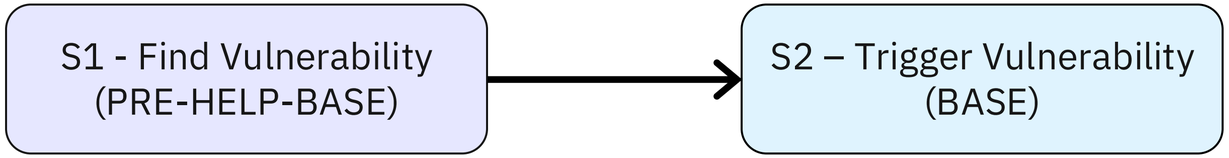
IBM

Exploitation Steps

Exploitation Steps

S1 - Find Vulnerability
(PRE-HELP-BASE)

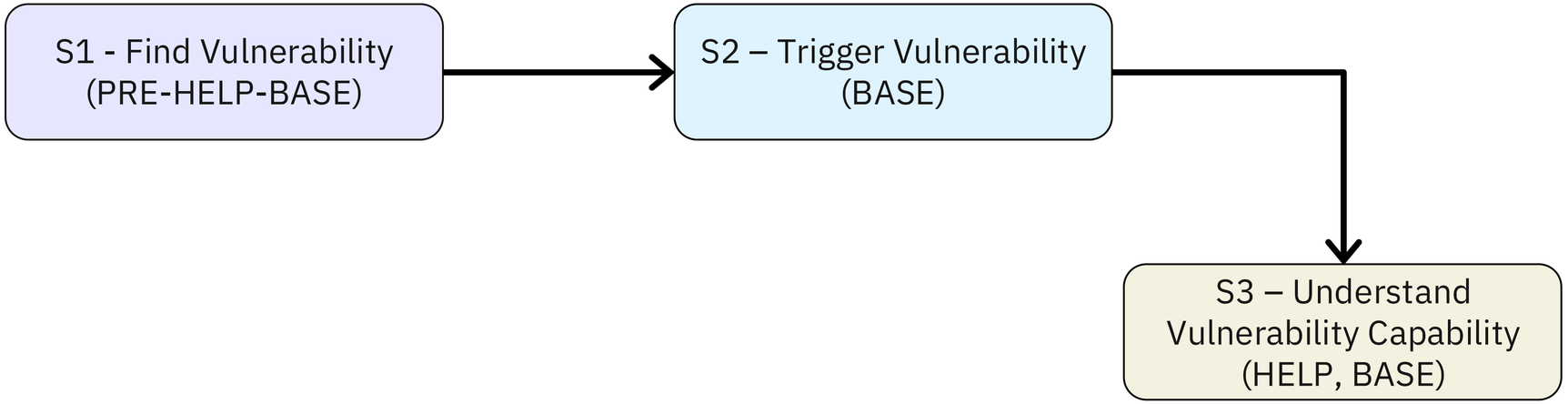
Exploitation Steps



armasuisse

IBM

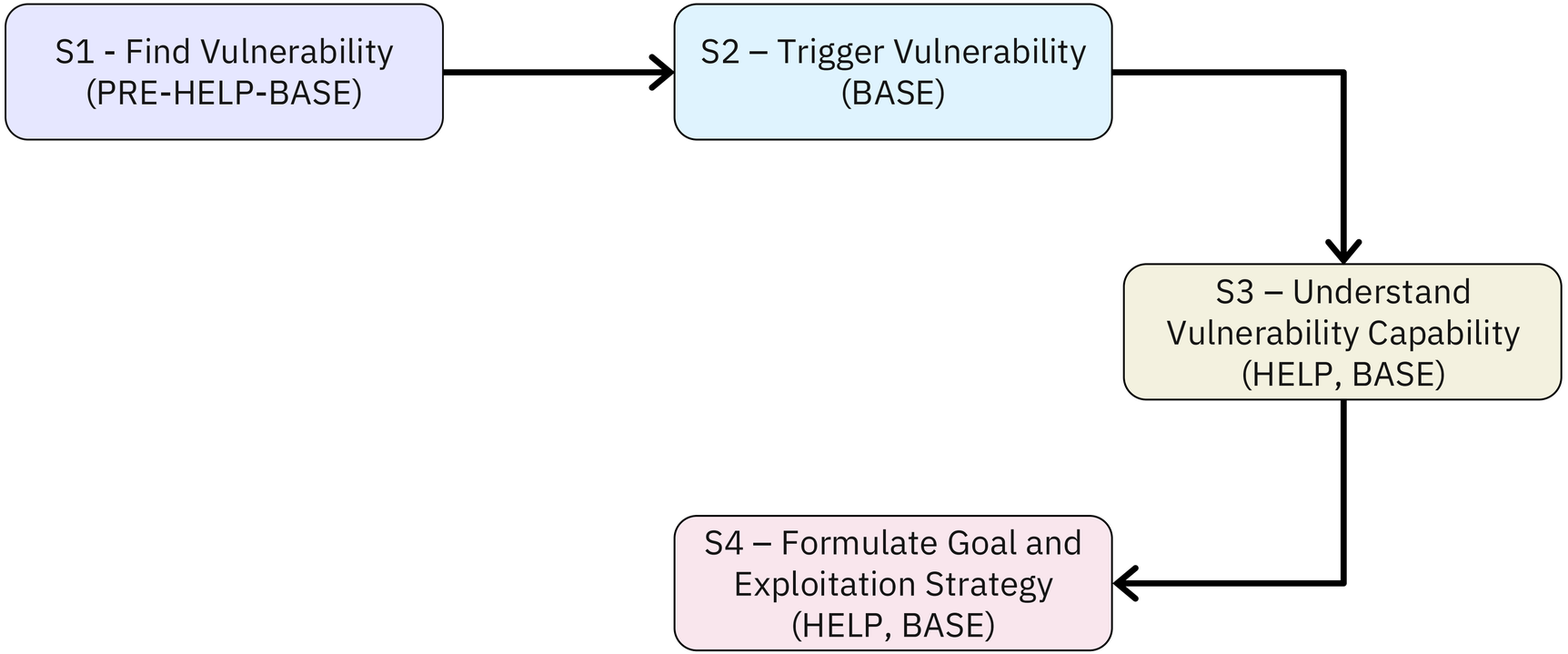
Exploitation Steps



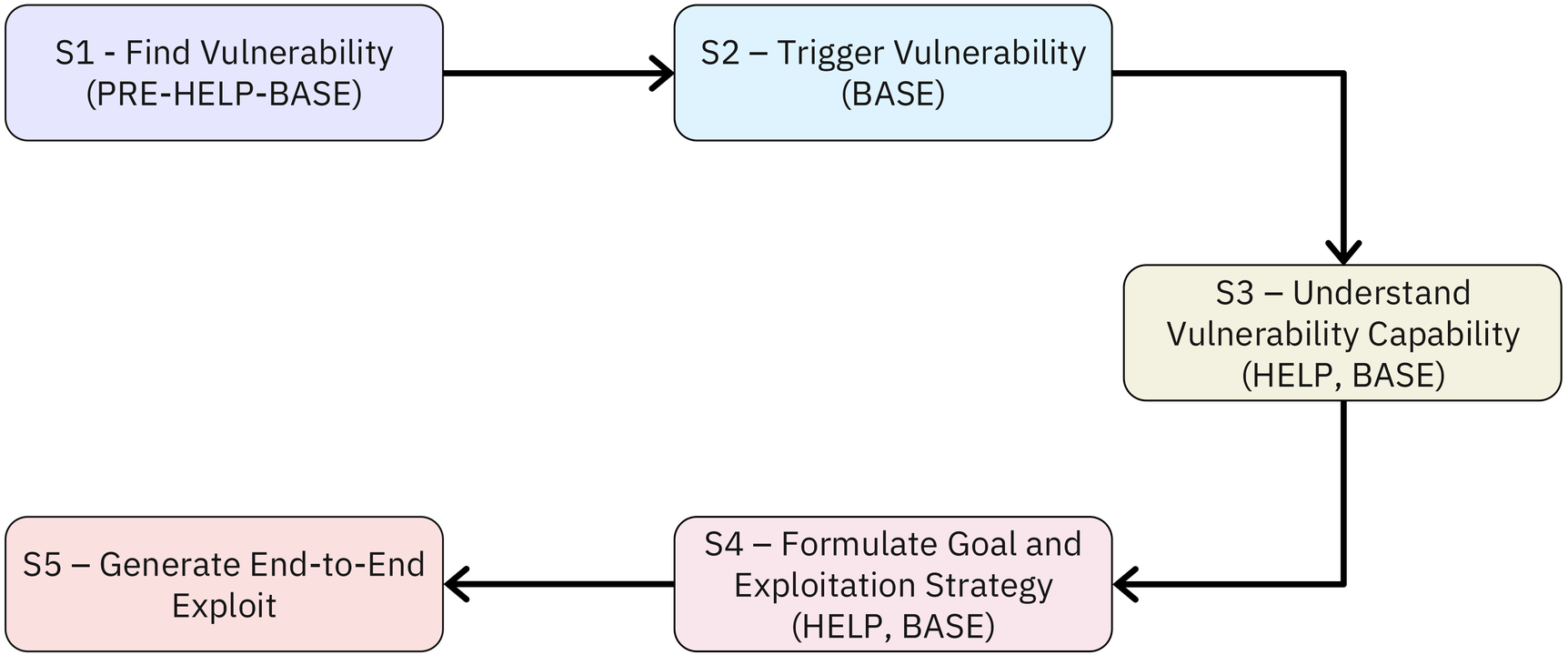
armasuisse

IBM

Exploitation Steps



Exploitation Steps



armasuisse

IBM

SoK – methodologies & statistics

First: **paper from top tier venues** from 2013 to 2023.

Second: **collected cited work** not included in first selection.

Filtered for **new techniques** or **automation** of some steps of **kernel exploitation**

SoK – methodologies & statistics

First: **paper from top tier venues** from 2013 to 2023.

Second: **collected cited work** not included in first selection.

Filtered for **new techniques** or **automation** of some steps of **kernel exploitation**

41 TOTAL

armasuisse

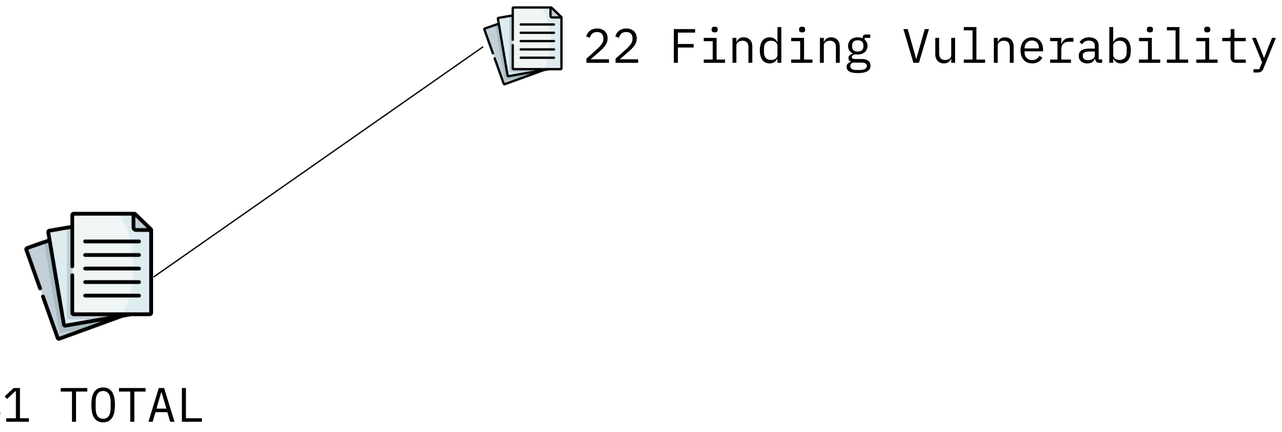
IBM

SoK – methodologies & statistics

First: **paper from top tier venues** from 2013 to 2023.

Second: **collected cited work** not included in first selection.

Filtered for **new techniques** or **automation** of some steps of **kernel exploitation**



armasuisse

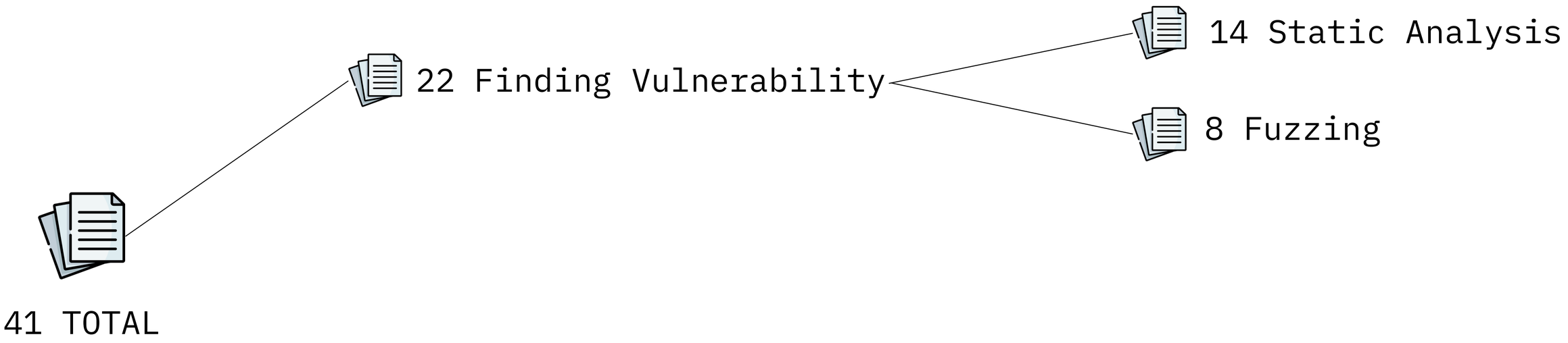
IBM

SoK – methodologies & statistics

First: **paper from top tier venues** from 2013 to 2023.

Second: **collected cited work** not included in first selection.

Filtered for **new techniques** or **automation** of some steps of **kernel exploitation**



armasuisse

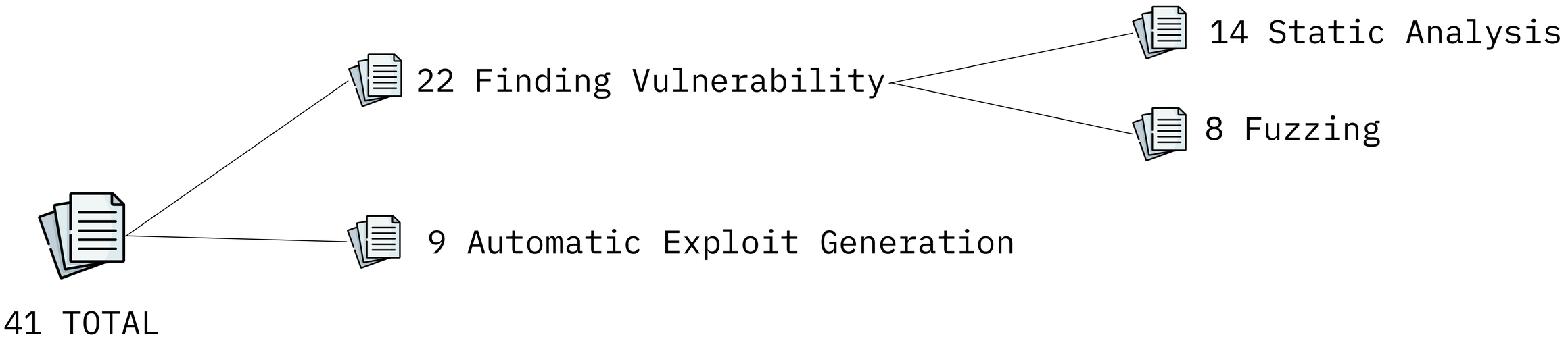
IBM

SoK – methodologies & statistics

First: **paper from top tier venues** from 2013 to 2023.

Second: **collected cited work** not included in first selection.

Filtered for **new techniques** or **automation** of some steps of **kernel exploitation**



armasuisse

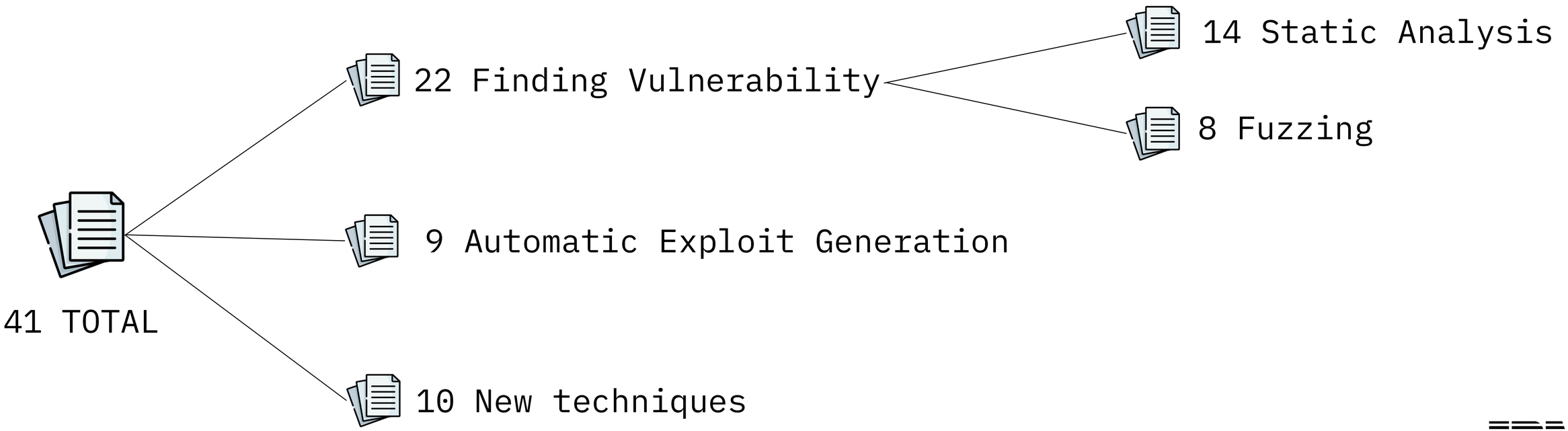
IBM

SoK – methodologies & statistics

First: **paper from top tier venues** from 2013 to 2023.

Second: **collected cited work** not included in first selection.

Filtered for **new techniques** or **automation** of some steps of **kernel exploitation**



armasuisse

IBM

Open problems – Attacker Model and Goals

Open problems – Attacker Model and Goals

Open problems – Attacker Model and Goals

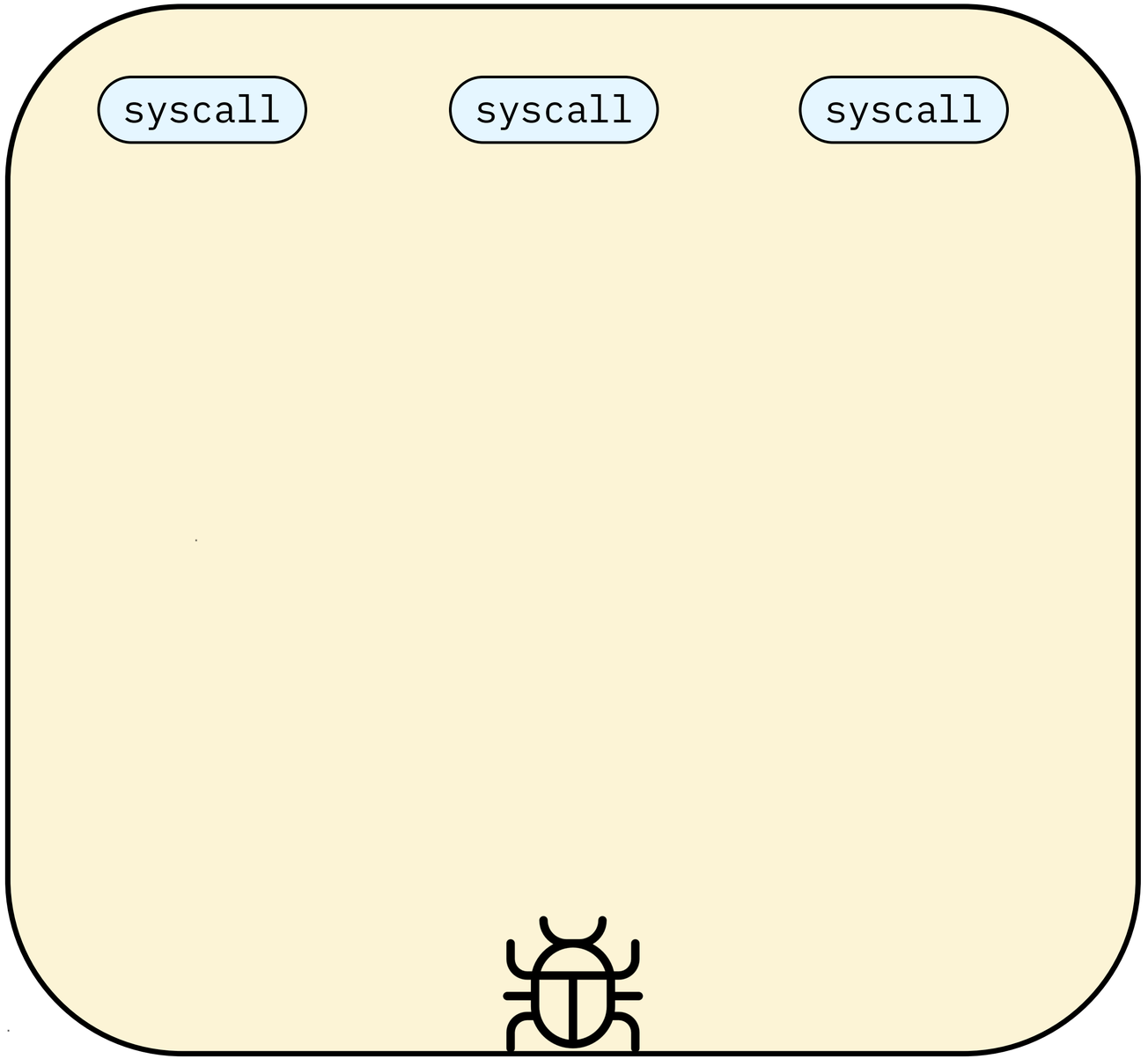
Most works are either threat model agnostic or focus solely to the local attacker scenario.

Open problems – Attacker Model and Goals

Most works are either threat model agnostic or focus solely to the local attacker scenario.

Attacker goals considered are only to PE or IL. DoS is tackled by only one paper

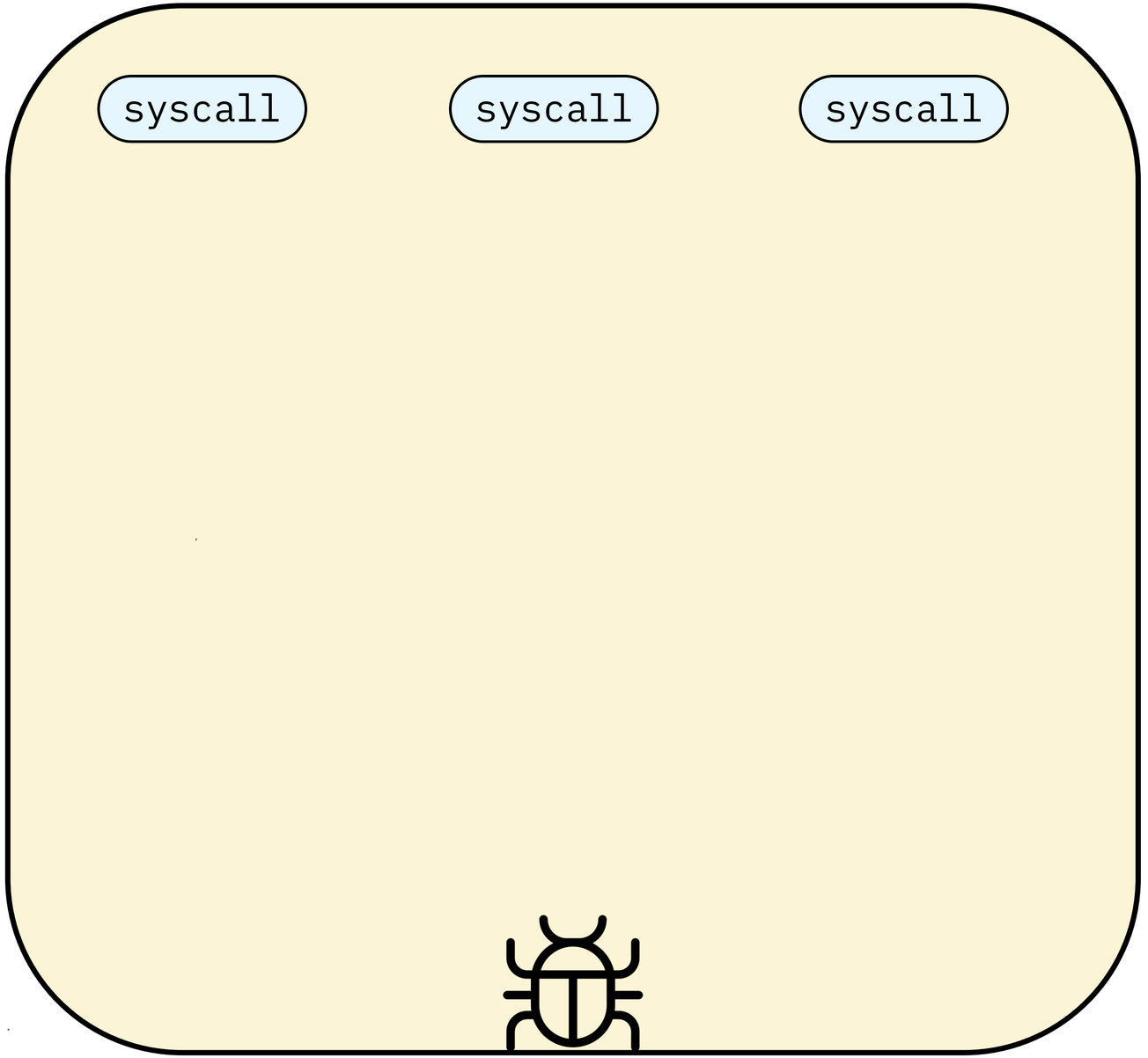
Open problems – Full Exploitability Assessment



armasuisse

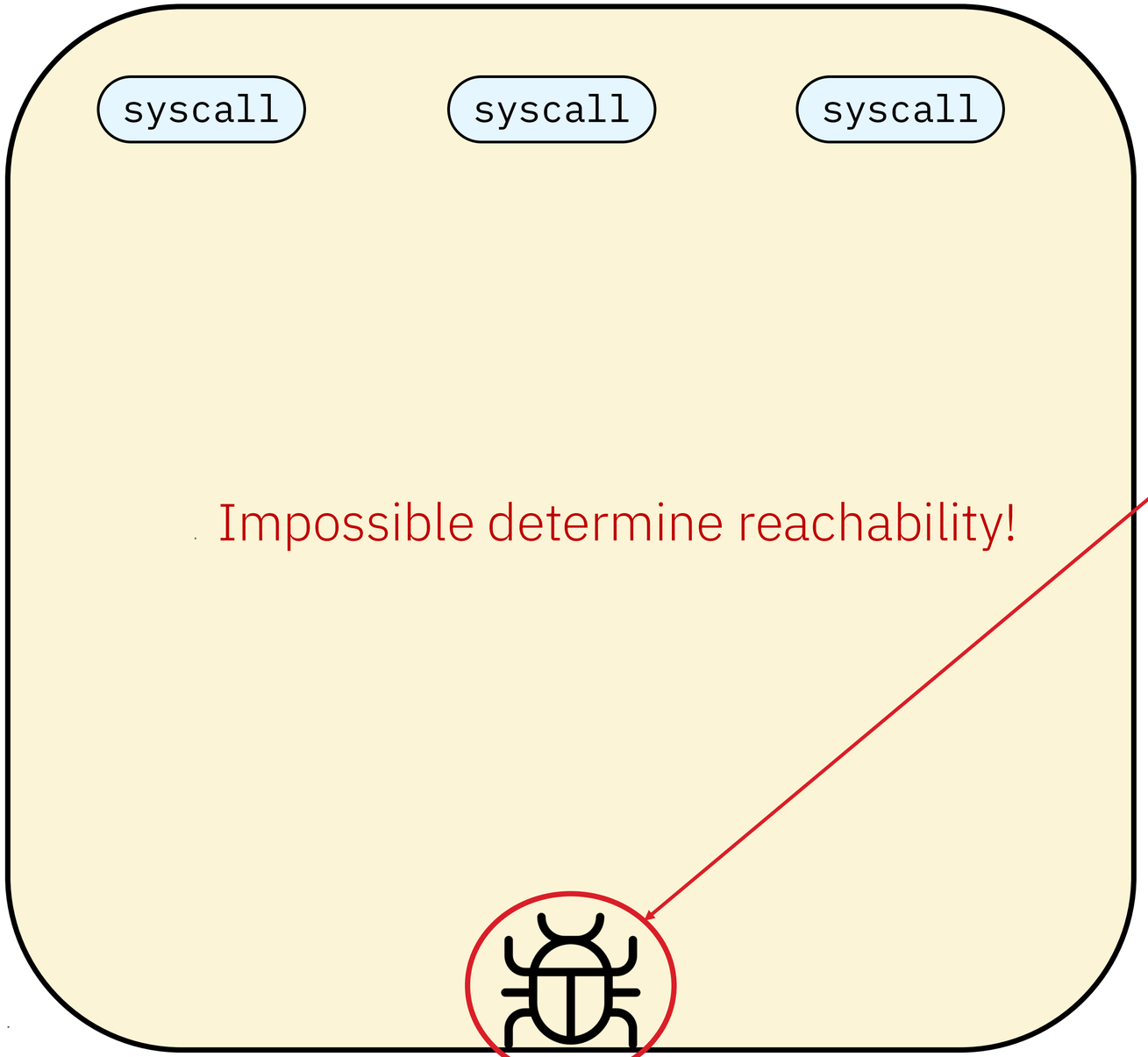
IBM

Open problems – Full Exploitability Assessment



static analysis

Open problems – Full Exploitability Assessment

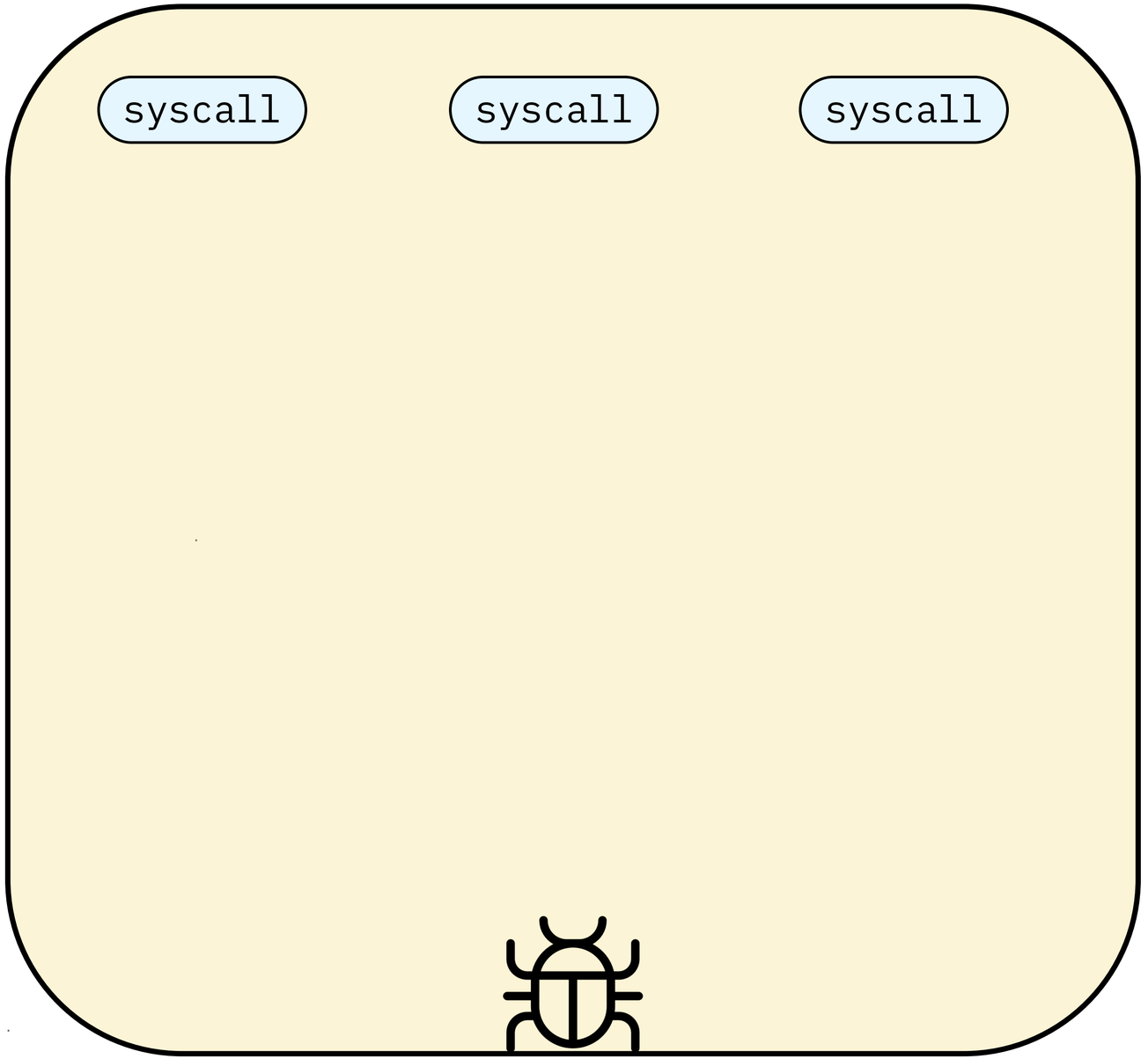


static analysis

armasuisse

IBM

Open problems – Full Exploitability Assessment

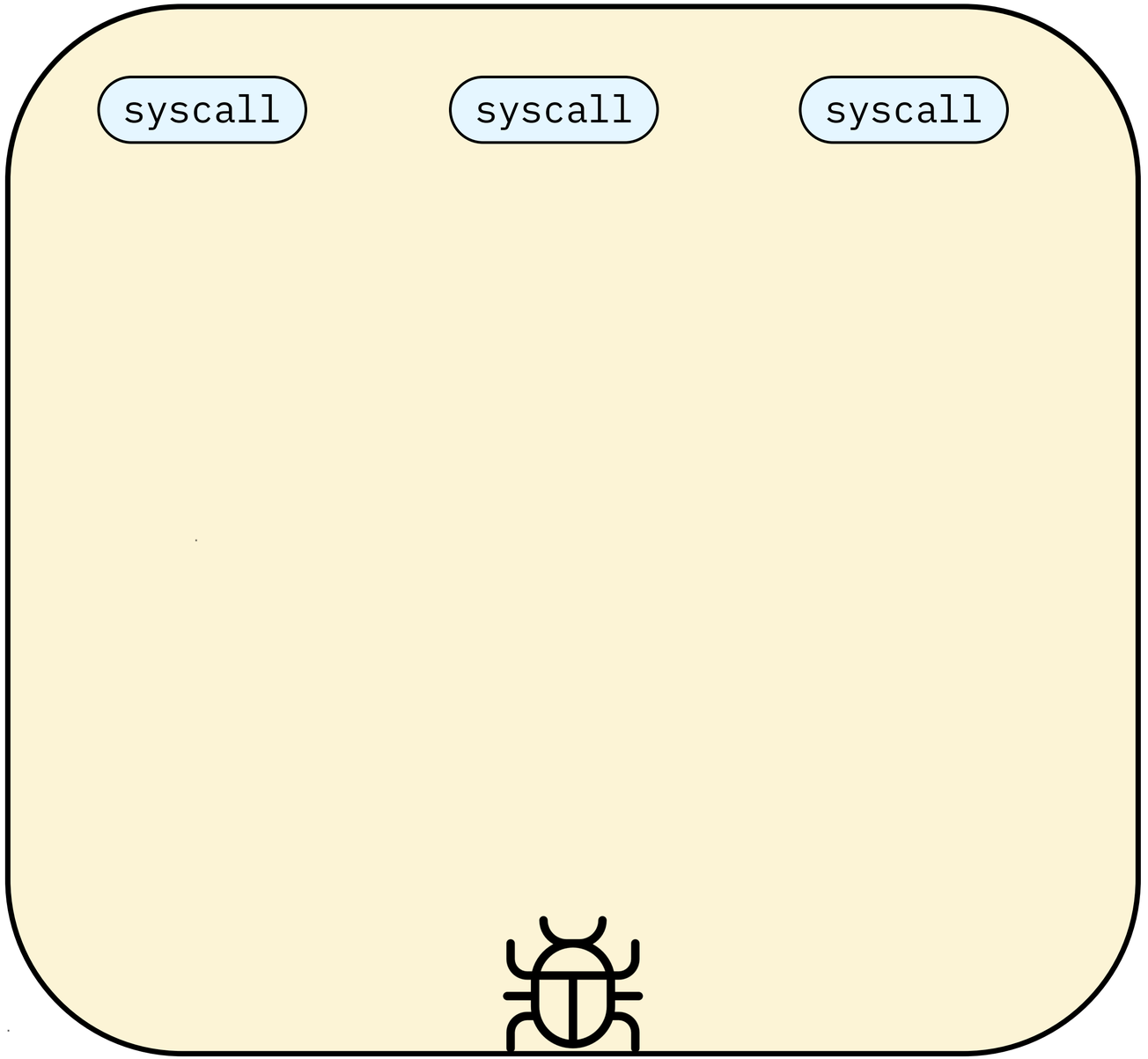


armasuisse

IBM

Open problems – Full Exploitability Assessment

dynamic analysis

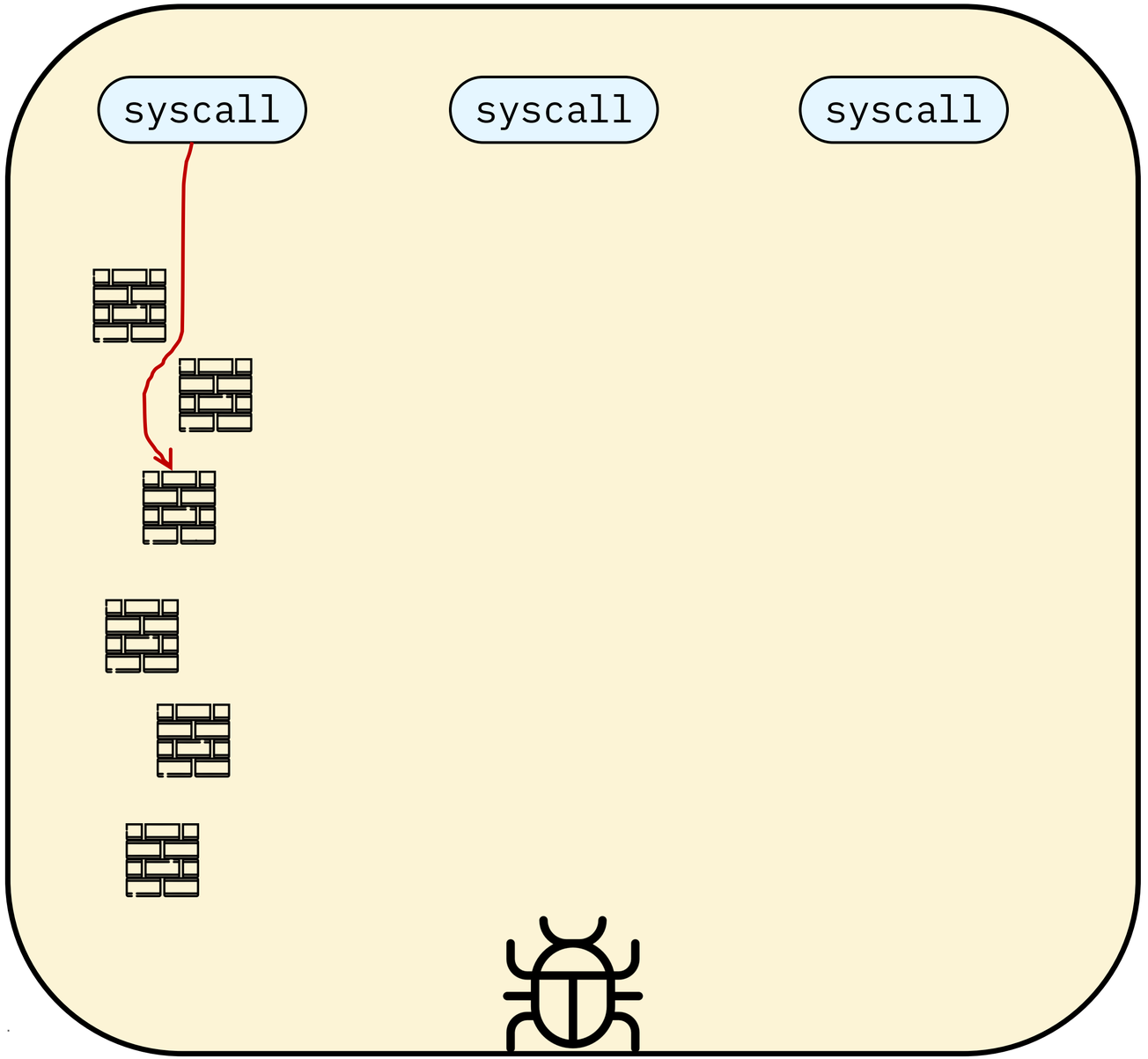


armasuisse

IBM

Open problems – Full Exploitability Assessment

dynamic analysis

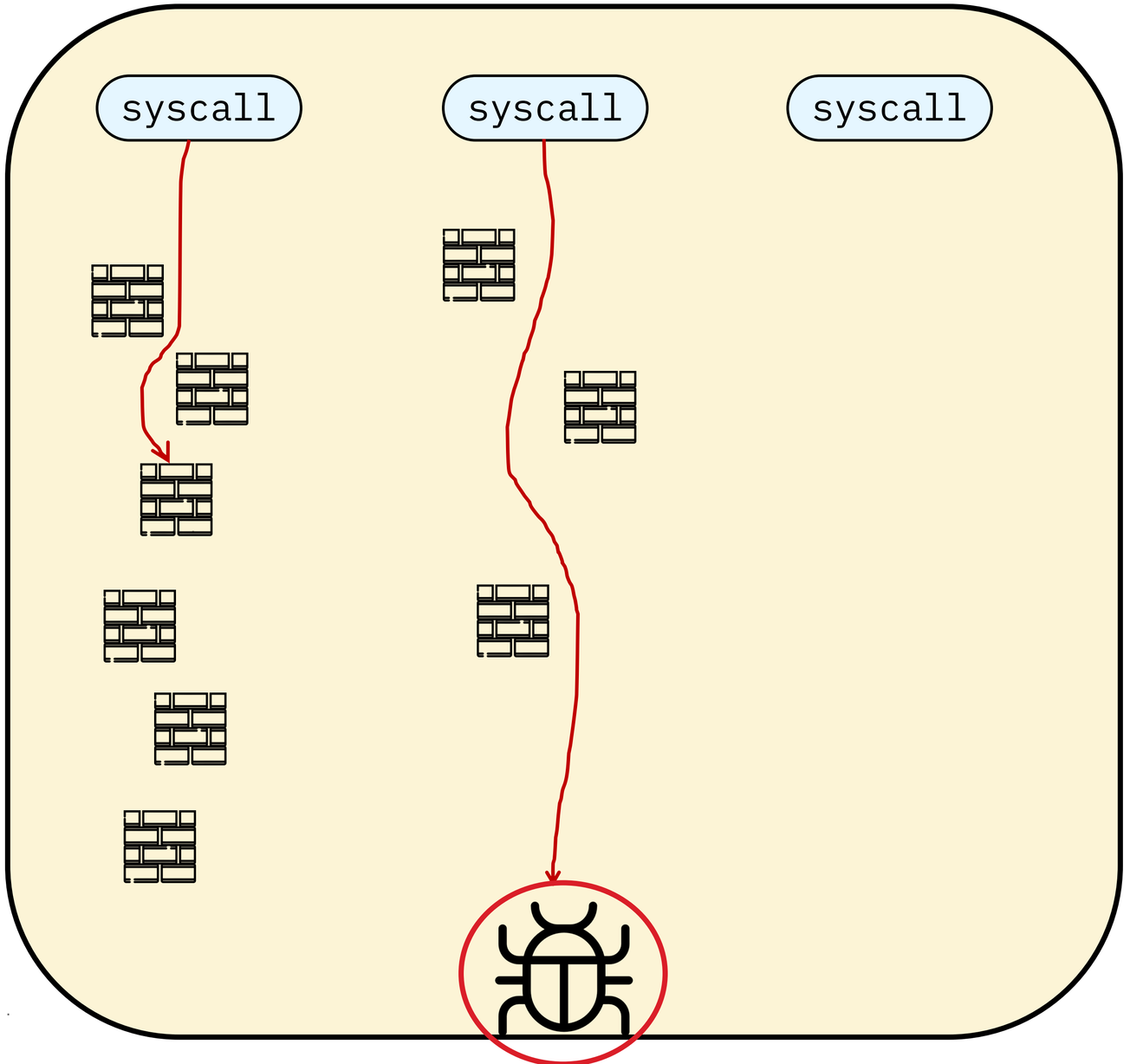


armasuisse

IBM

Open problems – Full Exploitability Assessment

dynamic analysis

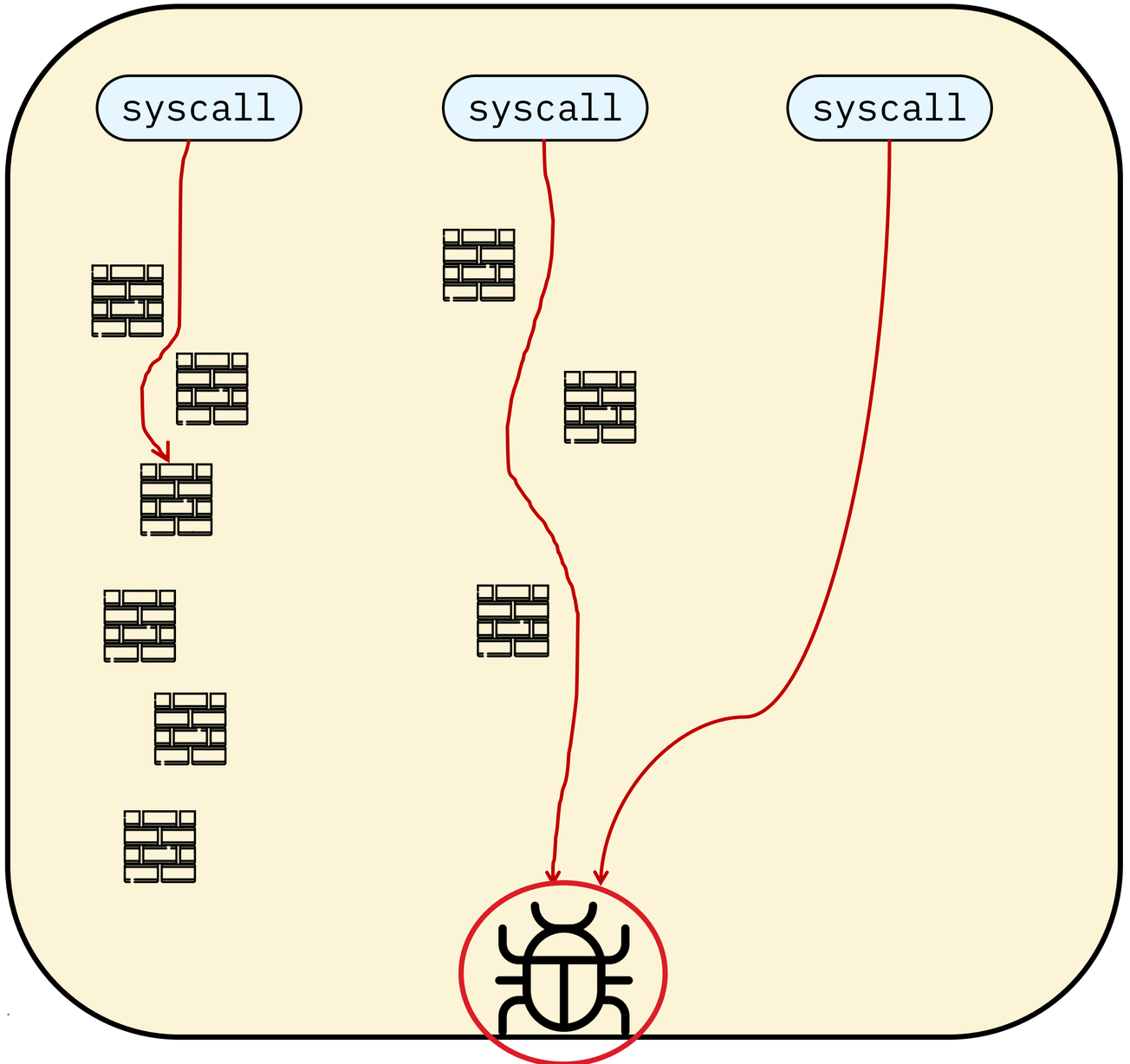


armasuisse

IBM

Open problems – Full Exploitability Assessment

dynamic analysis

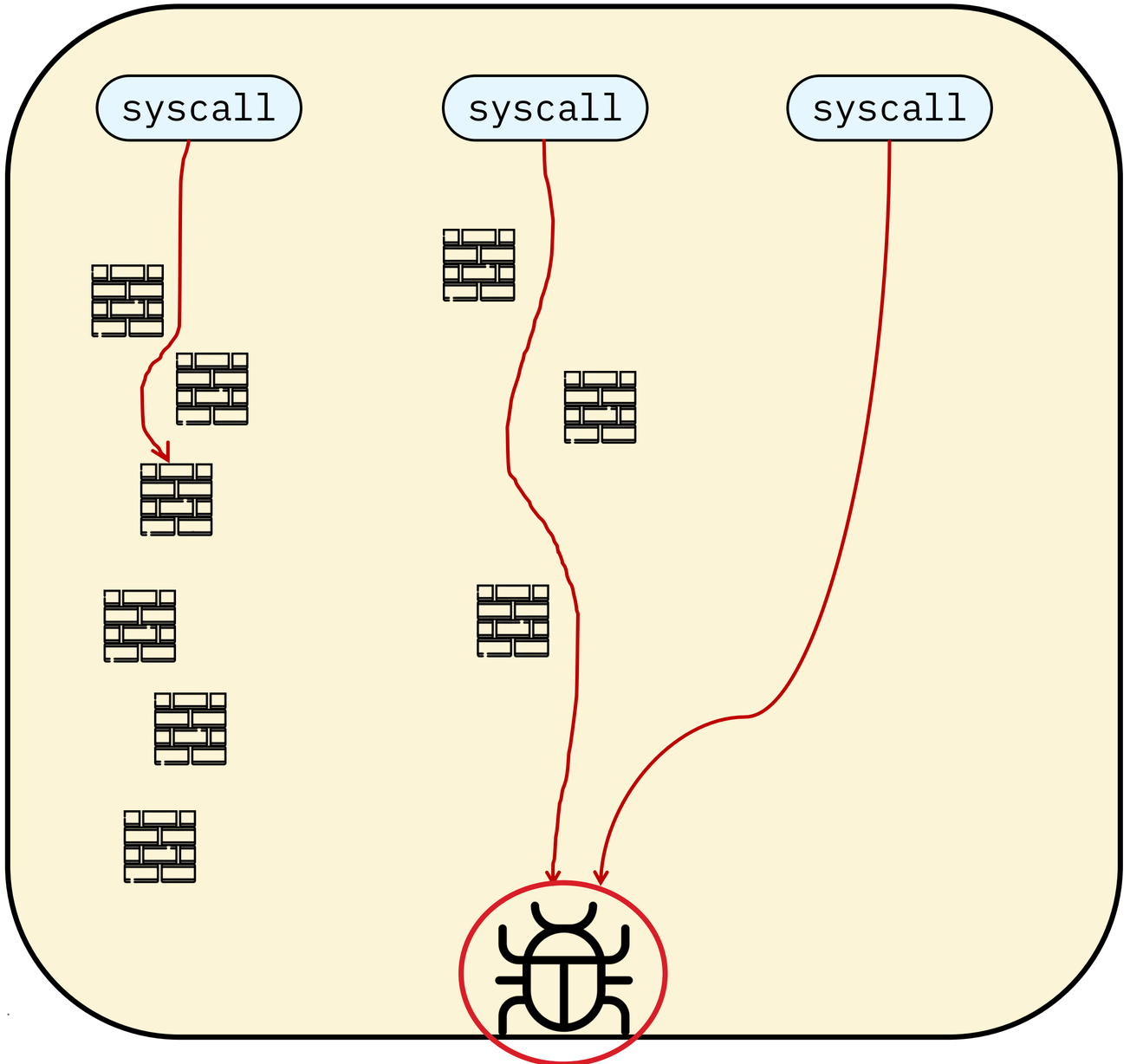


armasuisse

IBM

Open problems – Full Exploitability Assessment

dynamic analysis



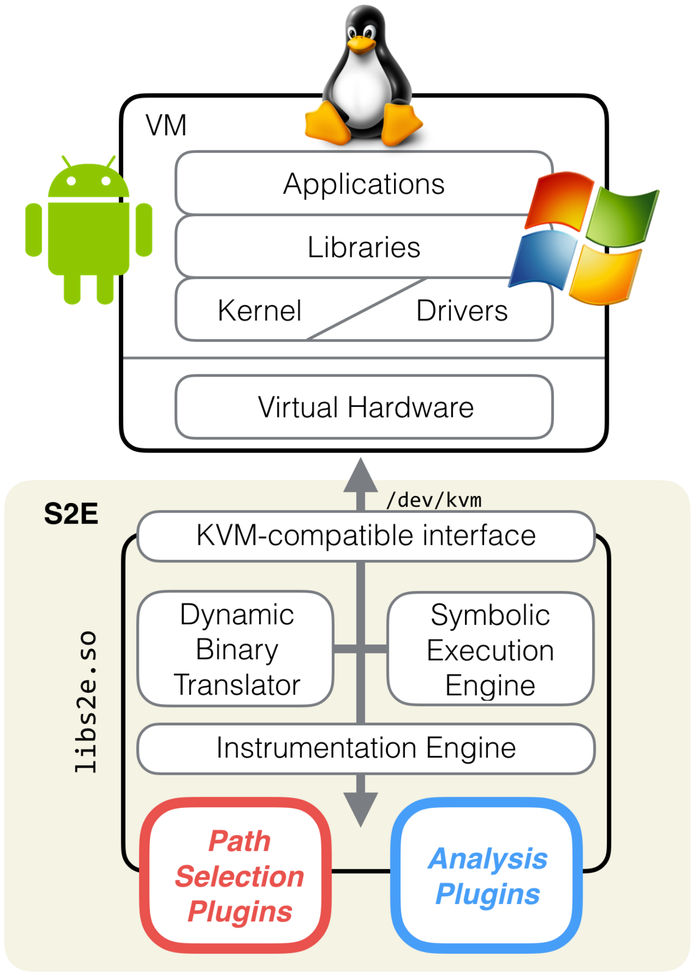
Only GREBE & SyzScope explore multiple/all paths to the same bug.

armasuisse

IBM

Open problems – Portability

Let’s consider the widely used S^{2}E as an example



Depends on:

1. Qemu v3.0.0 (mainline is at v10)
2. LLVM lifter
3. Klee
4. Instrumented kernel (version 4.9.3 until last year)
5. Custom KVM implementation

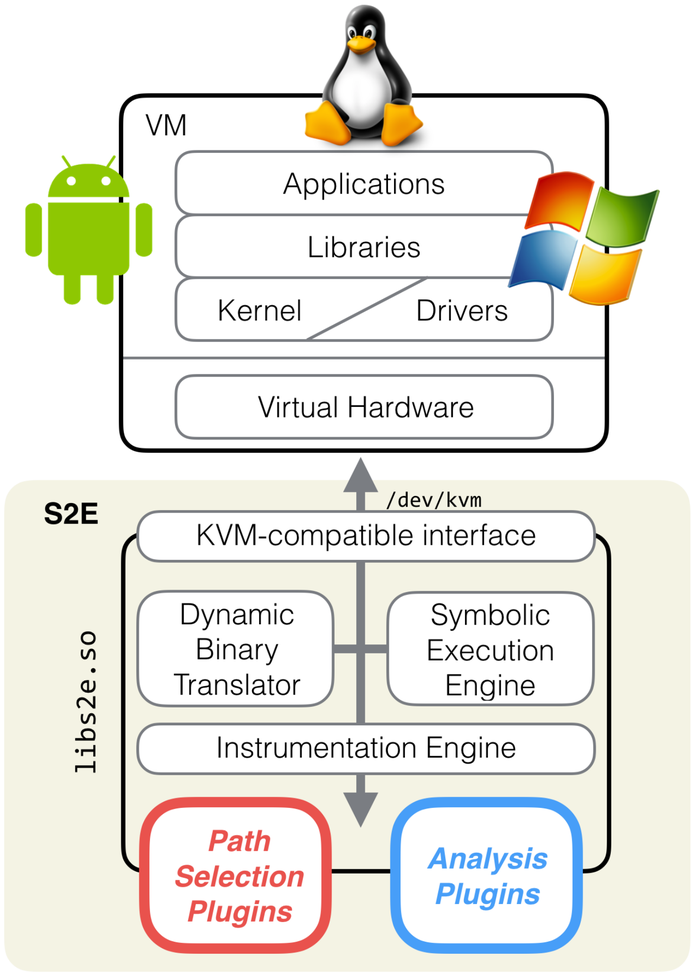
armasuisse

Image source: https://s2e.systems/

IBM

Open problems – Portability

Let’s consider the widely used S^{2}E as an example



Depends on:

1. Qemu v3.0.0 (mainline is at v10)
2. LLVM lifter
3. Klee
4. Instrumented kernel (version 4.9.3 until last year)
5. Custom KVM implementation

armasuisse

Image source: https://s2e.systems/

IBM

Open problems – Reproducibility

Reproducing and comparing with previous work is hard and requires expert knowledge

Each paper select their favorite set of bugs and kernel version

The community could agree on one research Linux version and bug dataset

The known bugs could be ported to the supported kernel automatically (difficult!)

Open problems – Tool reuse

We found that Syzkaller, LLVM, S2E and angr are overall highly used - kudos!

However, papers using these tools re-implemented within them similar features which are never merged into the tool mainline

Often, no code is provided

Methodologies from different papers often cannot be combined due to incompatible or brittle codebases

Conclusions - Status Kernel AEG

No work has yet achieved full AEG for the kernel at scale.

Exploitation steps are often tackled in isolation.

Focus is given to only the most prominent threat model and attacker goals.

Very limited re-use of state of the art beyond major analysis framework.