THALES
Building a future we can all trust

IETR

Formal Verification of the Heatseeker Technique for Detection of Code-Reuse Attacks

Téo Biton, Nikolai Kosmatov, Olivier Gilles, Daniel Gracia Pérez and Sébastien Pillement

www.thalesgroup.com



Context: Security of Computing Systems

> Computing systems face significant threats from software vulnerabilities

> Code-reuse attacks (CRAs)

- CRAs do not inject new code, but reuse existing code snippets ("gadgets")

> Jump-Oriented Programming (JOP) orchestrates an attack using jumps to gadgets

> JOP attacks use a dispatcher gadget (orchestrator) to control program flow

> Standard memory protections alone are insufficient; advanced detection strategies are necessary

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007
This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Code-Reuse Attacks: Intuition

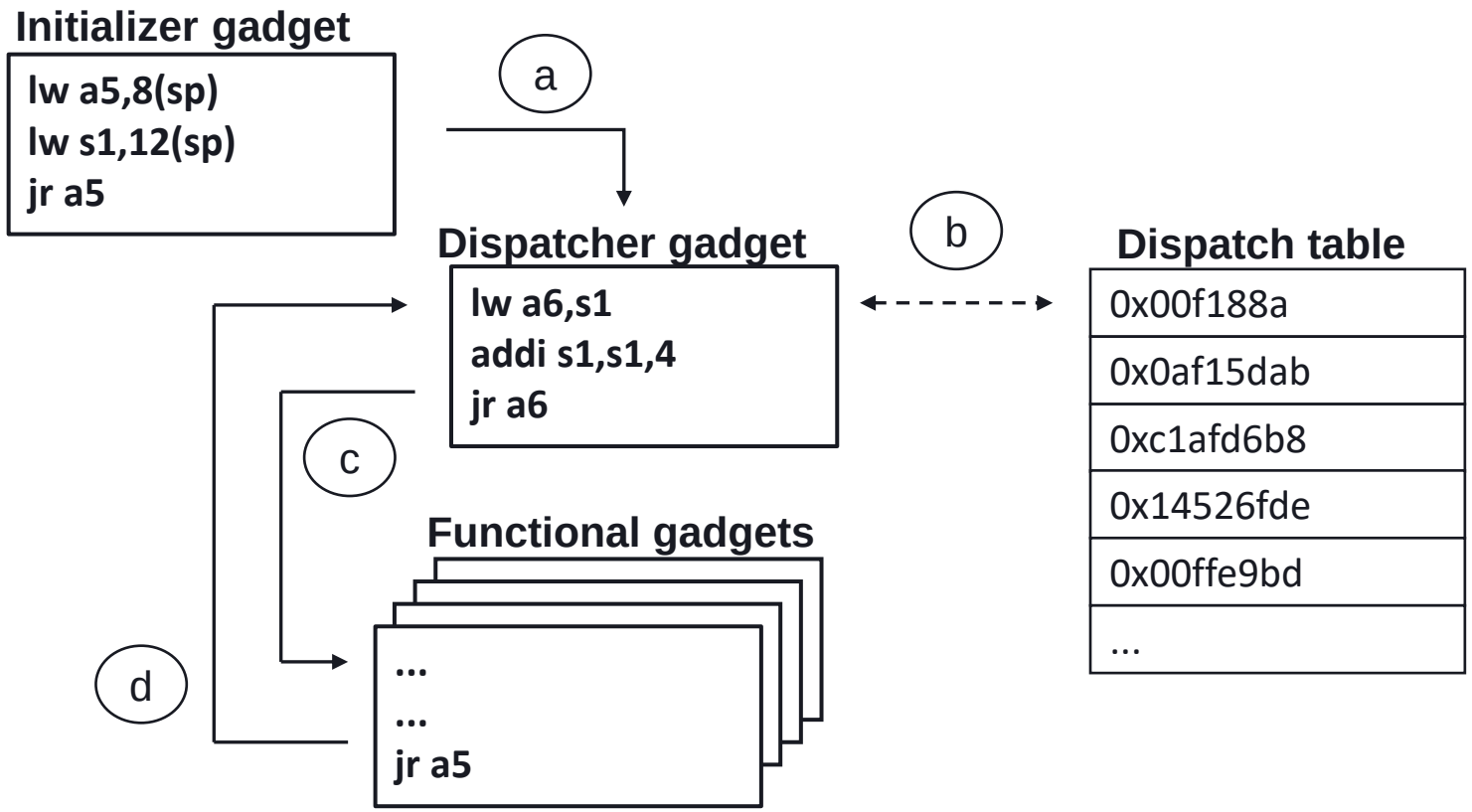
> CRAs rearrange existing pieces of code (called *gadgets*) in a new way in order to get a new behavior

> Like a ransom note rearranges pieces of text to get a new meaning



REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007
This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.
{THALES GROUP LIMITED DISTRIBUTION}

Jump-Oriented Programming Attack



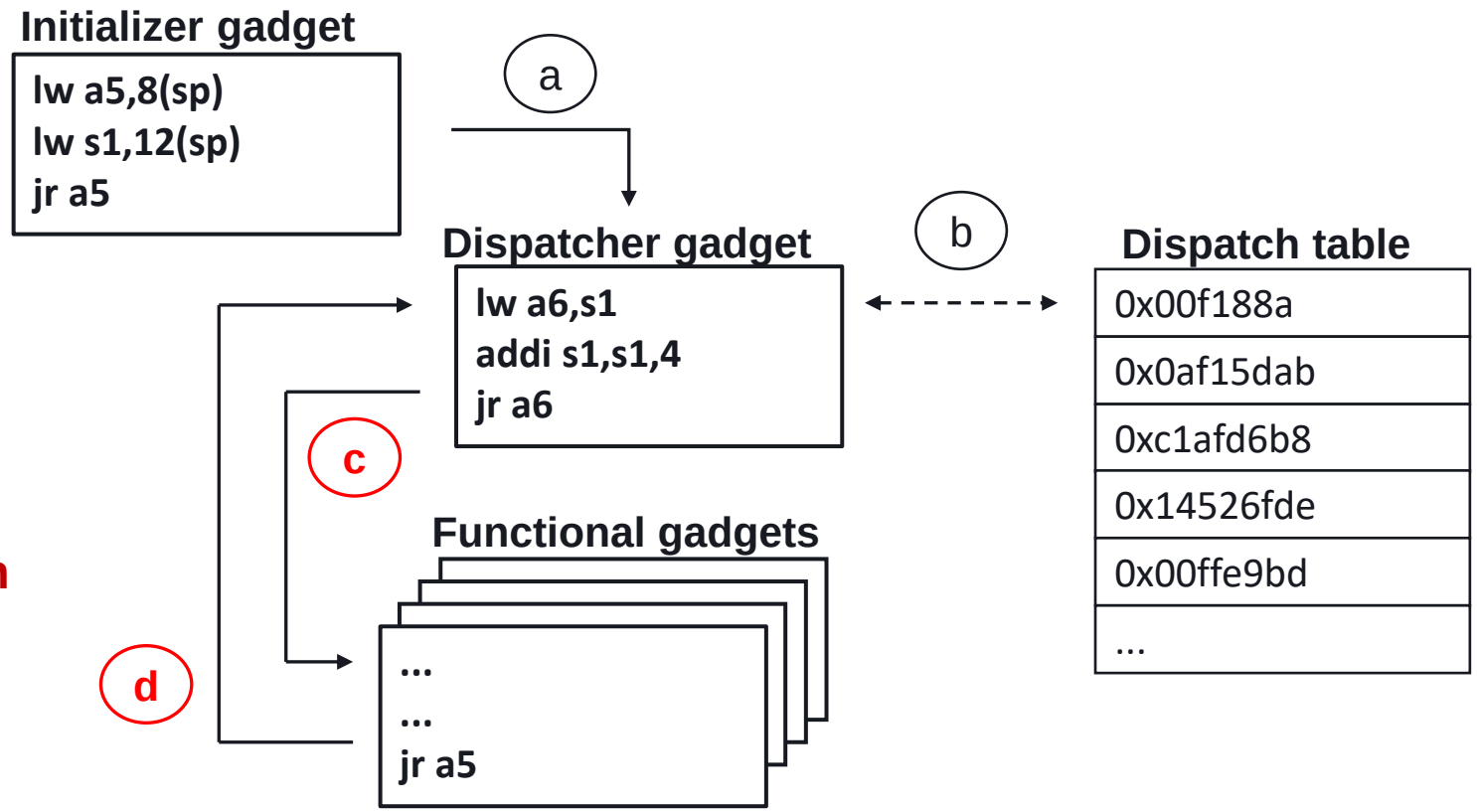
REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Jump-Oriented Programming Attack

- Repetition of specific patterns
- Jumps to and from the dispatcher gadget



REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Motivation: Ensuring Correctness of Attack Detection Mechanisms

- In previous work, we proposed the Heatseeker algorithm for dynamic hardware-based detection of JOP exploits [1]
- Signature-based detection technique based on attack patterns
- Target: Embedded systems
  - RISC-V architecture: CV32A6/CV64A6
- Our main goal: Prove Heatseeker's capability to detect JOP attacks

[1] Biton et al. Heatseeker: Uncovering Dispatcher Gadgets on the Fly. TACO 2026

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007
This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.
{THALES GROUP LIMITED DISTRIBUTION}

Overview of the Heatseeker Algorithm

> Characteristics of JOP exploits

- Gadgets are generally short (less than 5 instructions)
- The dispatcher gadget transfers control to functional gadgets
- The functional gadgets transfer control back to the dispatcher gadget

> Dynamically available data

- Source addresses, destination addresses and registers used by indirect jumps

> Heatseeker detects JOP attacks based on three criteria

- Redundance of source and destination addresses
- Redundant registers used by the jumps
- Short distance between these redundant addresses, forming the dispatcher gadget

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

The Heatseeker Algorithm: Two Caches

> Record indirect jumps (with source, destination, register) in two caches

> Dispatcher cache

- for likely jumps from the dispatcher gadget to various functional gadgets
- => same source (As), same register (R), various destinations (Ad)
- Count the number of pairs (As, R)

> Functional cache

- for likely jumps from a functional gadget to the dispatcher gadget
- => same destination (Ad), same register (R) or set of registers, various sources (As)
- Count the number of pairs (Ad, R)

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Example of a JOP Attack

As = address source, Ad = address destination, R = Register

Step	As	Ad	R	Transition
0	0x5B8C	0x3FAA	x6	$IG \rightarrow DG$
1	0x3FC0	0x7E50	x7	$DG \rightarrow FG_0$
2	0x7E72	0x3FAA	x6	$FG_0 \rightarrow DG$
3	0x3FC0	0x4BB8	x7	$DG \rightarrow FG_1$
4	0x4BE4	0x3FAA	x6	$FG_1 \rightarrow DG$
5	0x3FC0	0x12D4	x7	$DG \rightarrow FG_2$

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

The Heatseeker Algorithm: Jump Monitoring

likely jumps from a functional gadget to the dispatcher gadget

likely jumps from the dispatcher gadget to a functional gadget

Step	As	Ad	R	Transition
0	0x5B8C	0x3FAA	x6	$IG \rightarrow DG$
1	0x3FC0	0x7E50	x7	$DG \rightarrow FG_0$
2	0x7E72	0x3FAA	x6	$FG_0 \rightarrow DG$
3	0x3FC0	0x4BB8	x7	$DG \rightarrow FG_1$
4	0x4BE4	0x3FAA	x6	$FG_1 \rightarrow DG$
5	0x3FC0	0x12D4	x7	$DG \rightarrow FG_2$

	Functional cache				*Dispatcher cache*			
Entry	R	Ad	As	counter	R	As	Ad	counter
0	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
1	—	—	—	—	—	—	—	—
2	—	—	—	—	—	—	—	—
0	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
1	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
2	—	—	—	—	—	—	—	—
0	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
1	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
2	—	—	—	—	6	0x5B8C	0x3FAA	1
0	7	0x4BB8	0x3FC0	1	**7**	**0x3FC0**	**0x4BB8**	**2**
1	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
2	7	0x7E50	0x3FC0	1	6	0x5B8C	0x3FAA	1

ll rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

The Heatseeker Algorithm: Jump Monitoring

likely jumps from a functional gadget to the dispatcher gadget

likely jumps from the dispatcher gadget to a functional gadget

Step	As	Ad	R	Transition
0	0x5B8C	0x3FAA	x6	$IG \rightarrow DG$
1	0x3FC0	0x7E50	x7	$DG \rightarrow FG_0$
2	0x7E72	0x3FAA	x6	$FG_0 \rightarrow DG$
3	0x3FC0	0x4BB8	x7	$DG \rightarrow FG_1$
4	0x4BE4	0x3FAA	x6	$FG_1 \rightarrow DG$
5	0x3FC0	0x12D4	x7	$DG \rightarrow FG_2$

	Functional cache				*Dispatcher cache*			
Entry	R	Ad	As	counter	R	As	Ad	counter
0	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
1	—	—	—	—	—	—	—	—
2	—	—	—	—	—	—	—	—
0	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
1	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
2	—	—	—	—	—	—	—	—
0	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
1	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
2	—	—	—	—	6	0x5B8C	0x3FAA	1
0	7	0x4BB8	0x3FC0	1	**7**	**0x3FC0**	**0x4BB8**	**2**
1	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
2	7	0x7E50	0x3FC0	1	6	0x5B8C	0x3FAA	1

ll rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

The Heatseeker Algorithm: Jump Monitoring

likely jumps from a functional gadget to the dispatcher gadget

likely jumps from the dispatcher gadget to a functional gadget

Step	As	Ad	R	Transition
0	0x5B8C	0x3FAA	x6	$IG \rightarrow DG$
1	0x3FC0	0x7E50	x7	$DG \rightarrow FG_0$
2	0x7E72	0x3FAA	x6	$FG_0 \rightarrow DG$
3	0x3FC0	0x4BB8	x7	$DG \rightarrow FG_1$
4	0x4BE4	0x3FAA	x6	$FG_1 \rightarrow DG$
5	0x3FC0	0x12D4	x7	$DG \rightarrow FG_2$

	Functional cache				*Dispatcher cache*			
Entry	R	Ad	As	counter	R	As	Ad	counter
0	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
1	—	—	—	—	—	—	—	—
2	—	—	—	—	—	—	—	—
0	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
1	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
2	—	—	—	—	—	—	—	—
0	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
1	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
2	—	—	—	—	6	0x5B8C	0x3FAA	1
0	7	0x4BB8	0x3FC0	1	**7**	**0x3FC0**	**0x4BB8**	**2**
1	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
2	7	0x7E50	0x3FC0	1	6	0x5B8C	0x3FAA	1

ll rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

The Heatseeker Algorithm: Jump Monitoring

likely jumps from a functional gadget to the dispatcher gadget

likely jumps from the dispatcher gadget to a functional gadget

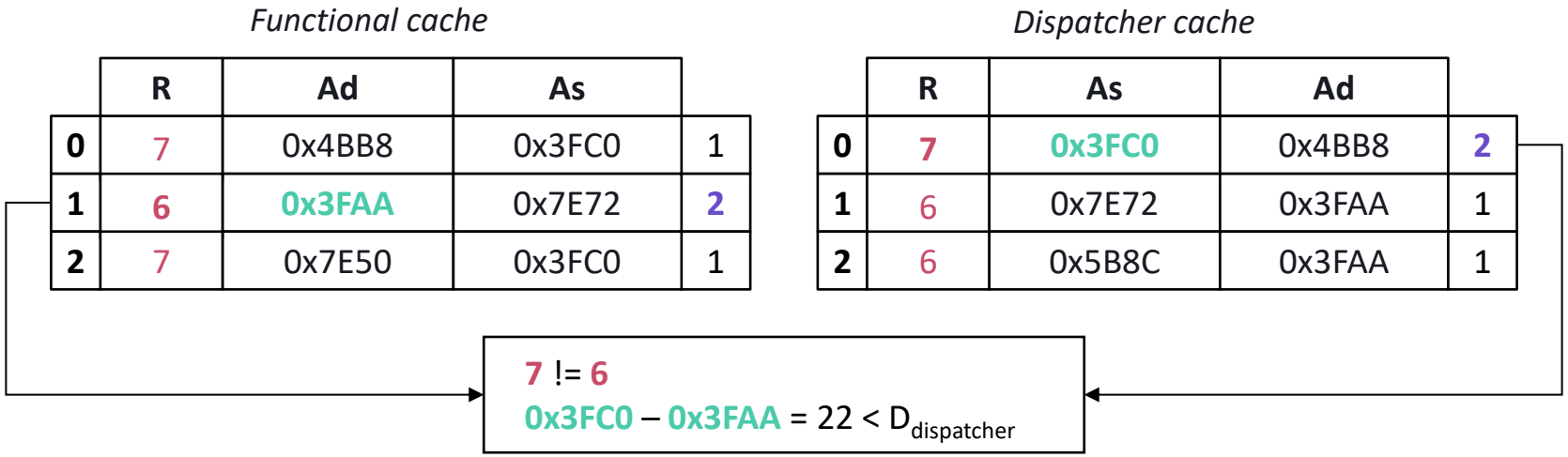
Step	As	Ad	R	Transition
0	0x5B8C	0x3FAA	x6	$IG \rightarrow DG$
1	0x3FC0	0x7E50	x7	$DG \rightarrow FG_0$
2	0x7E72	0x3FAA	x6	$FG_0 \rightarrow DG$
3	0x3FC0	0x4BB8	x7	$DG \rightarrow FG_1$
4	0x4BE4	0x3FAA	x6	$FG_1 \rightarrow DG$
5	0x3FC0	0x12D4	x7	$DG \rightarrow FG_2$

	Functional cache				*Dispatcher cache*			
Entry	R	Ad	As	counter	R	As	Ad	counter
0	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
1	—	—	—	—	—	—	—	—
2	—	—	—	—	—	—	—	—
0	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
1	6	0x3FAA	0x5B8C	1	6	0x5B8C	0x3FAA	1
2	—	—	—	—	—	—	—	—
0	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
1	7	0x7E50	0x3FC0	1	7	0x3FC0	0x7E50	1
2	—	—	—	—	6	0x5B8C	0x3FAA	1
0	7	0x4BB8	0x3FC0	1	**7**	**0x3FC0**	**0x4BB8**	**2**
1	**6**	**0x3FAA**	**0x7E72**	**2**	6	0x7E72	0x3FAA	1
2	7	0x7E50	0x3FC0	1	6	0x5B8C	0x3FAA	1

ll rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Illustration of the Heatseeker Algorithm: Detection



- We observed 2 likely jumps to a dispatcher gadget (same destination, same register)
- We observed 2 likely jumps from a dispatcher gadget (same source, same register)
- The registers are different
- The size of the likely dispatcher gadget is below the chosen max. dispatcher gadget size D
- Parameters: number of cache lines, suspicious number of jumps G, max.dispatcher gadget size D, number of attack registers used by functional gadgets, etc.

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007
This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.
{THALES GROUP LIMITED DISTRIBUTION}

Proof of the Heatseeker Algorithm: Overview

> Main goal: proof of the capacity to detect JOP attacks, but also:

- Analysis of limitations of the algorithm: the kinds of attacks that are detected, under which assumptions (number of cache lines, number of registers used by attackers, relationships between cache entries…)
- Concrete counter-examples for non-proven cases

> Approach

- We formalize the algorithm as a C model with clearly stated assumptions and assertions
- We formalize an attack sequence by assumptions on the input sequence of jumps
- We apply CBMC, a well-known bounded model-checker for C programs
- While the proof with CBMC is bounded, the result is proved for any state and execution length

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Proof of the Heatseeker algorithm: Main Theorem

$\mathcal{P}_{\text{main}}$: Consider the heatseeker algorithm implemented with N cache lines, and consider JOP attack sequences in which $NumOp > 0$ denotes the number of steps, $D > 0$ a maximal dispatcher start-end distance, $G > 0$ the number of different gadgets, and set $MaxR = \lfloor N/2 \rfloor$. For any execution containing a JOP attack sequence with the considered constraints, the heatseeker algorithm effectively detects an attack by reporting an alarm.

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Proof of the Heatseeker algorithm: Methodology

- **Define an initial state Init**
- **Define an invariant I that is expected to hold in any reachable state of the algorithm**
- **Prove three key properties:**

$\mathcal{P}_1$: The initial state *Init* verifies $\mathcal{I}$.
$\mathcal{P}_2$: If the algorithm starts in a state verifying $\mathcal{I}$ and executes any step, the resulting state will also verify $\mathcal{I}$. In other words, $\mathcal{I}$ is indeed an invariant.
$\mathcal{P}_3$: Starting from a state that verifies $\mathcal{I}$, the execution for an attack sequence always leads to reporting an attack.

- **Proof justification:** By P1 and P2, when starting from the initial state, we always remain in a state respecting I. By P3, we always detect an attack starting in a state respecting I. Therefore, we detect an attack sequence starting at any reachable state.

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007
This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Proof of the Heatseeker Algorithm: Invariants

- $\mathcal{I}_1$ constrains the numbers of recorded lines Fi and Di in caches $\mathcal{F}$ and $\mathcal{D}$ to be in $[0, N]$, where N denotes the number of entries (lines) in a cache.
- $\mathcal{I}_2$ states that each non-recorded cache line (i.e. for index $i \geq Fi$ in $\mathcal{F}$ and for index $i \geq Di$ in $\mathcal{D}$) has default (zero) values.
- $\mathcal{I}_3$ states that each recorded cache line (i.e. for index $i < Fi$ in $\mathcal{F}$ and for index $i < Di$ in $\mathcal{D}$) has non-null tag address and tag register.
- $\mathcal{I}_4$ states that in each cache, each recorded cache line has a unique tag. In other words, two recoded lines in the same cache cannot share the same tag.
- $\mathcal{I}_5$ states that each recorded cache line has a gadget counter in $[1, G]$.

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Proof Statistics and Effort

> CBMC verification times for different parameter configurations:

N	$MaxR$	$NumOp$	Time (s)
3	1	5	1.002
4	2	9	3.543
5	2	11	8.214
6	3	15	32.216

> Effort: 4 person-months

- 2 person-months to create a model,
- 2 person-months to debug the proof and anylyze counter-examples

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Understanding of the Algorithm and its Limitations

- CBMC generated counter-examples for an earlier version of the model
- These counter-examples are spurious: they do not correspond to real-life attacks
- We added additional assumptions on attack scenarios to avoid the detection of such unrealistic attacks

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}

Conclusion and Future Work

- A formal model of JOP attacks and the Heatseeker detection algorithm
- Proof that the algorithm detects JOP attacks with CBMC for several configurations
- Better understanding of limitations
- A rigorous hardware defense ensured by formal verification on a model
- Companion artifact available at https://doi.org/10.5281/zenodo.19794964

Future work

- Extending the formal model to cover more complex JOP attacks
- Construction of general models that enable the detection of larger CRA variants
- Experiments with other verification tools

REF 062025/tbiton rev 000 – 05/06/2025 Thales / Template: 87211168-DOC-GRP-EN-007

This document may not be reproduced, modified, adapted, published, translated, in any way, in whole or in part or disclosed to a third party without the prior written consent of Thales © 2023 THALES. All rights reserved.

{THALES GROUP LIMITED DISTRIBUTION}