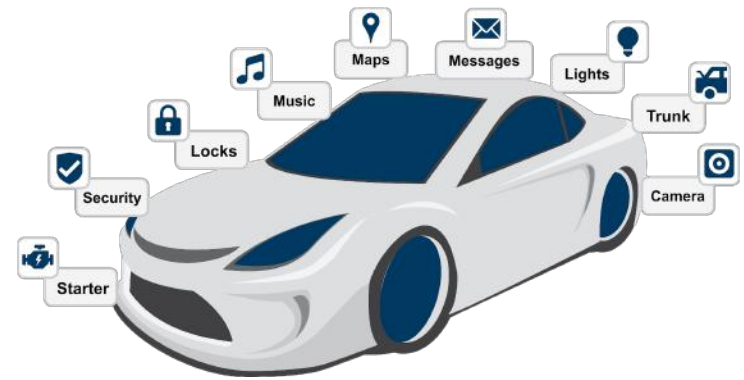
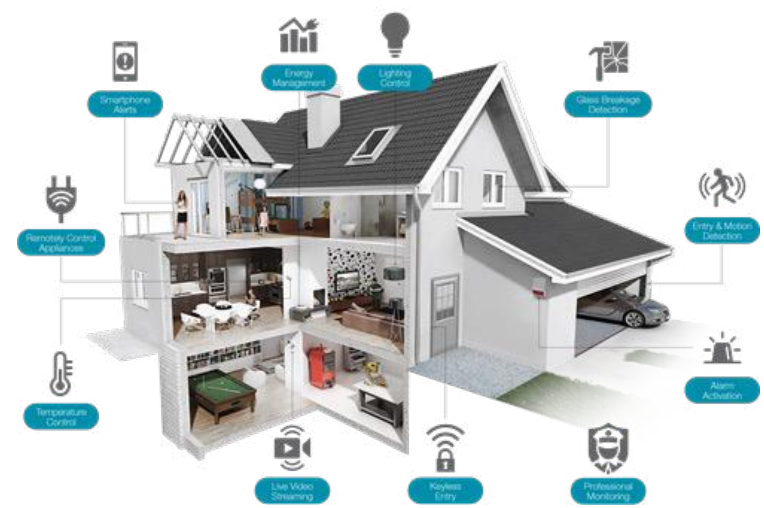
A software-based approach to secure bare-metal devices

Bruno Crispo

Michele Grisafi, Marco Roveri, Mahmoud Ammar, Bart Jacobs, Danny Hughes.

Inter-Connected devices

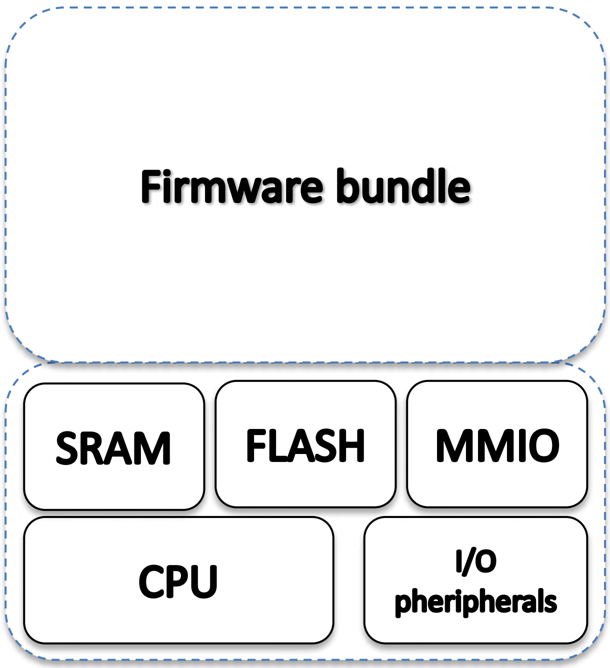




Motivations

In this context a purely software solution is a necessity

Bare-metal device



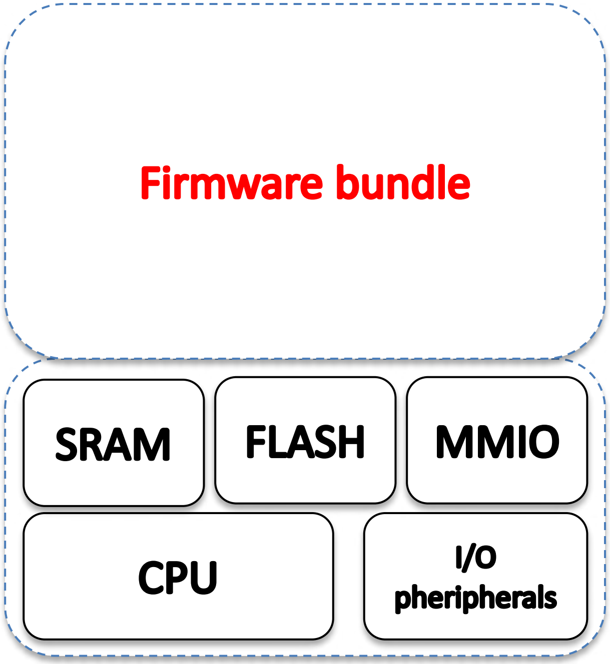
- No memory protection
  - No MMU
    - no ASLR, DEP, etc.
  - No MPU
- No hardcoded dual mode

- Typically few kB of Flash (˜100 kB) and RAM (˜10 kB)
- All AVR MCUs, some MSP430, some ARM Cortex-M

Trust model

SW supply-chain issues

Bare-metal device



- Developed (partially) by third-parties
- Rarely the open-source is available

Adversarial model

- Software remote-only attacks.
  - Tampering with any unprotected memory area.
  - Eavesdropping on communication
  - Inject malicious logic in existing applications
  - ….

- Availability and physical attacks are out of scope.

PISTIS: SW-based Trusted Computing Architecture

- Confidentiality and Integrity

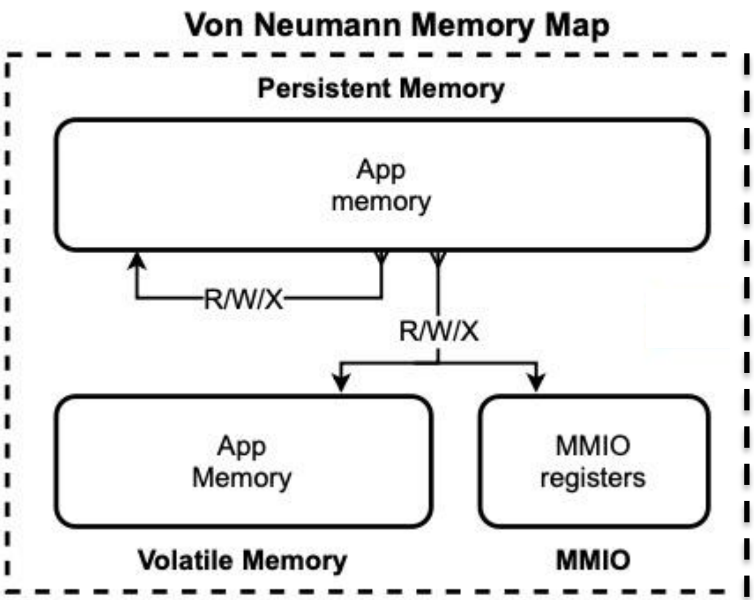
- Memory isolation technique based on selective software virtualization and assembly-level code verification
- Formally verified that the design preserve memory isolation
- Implementation (code) verified to be memory-safe and crash-free

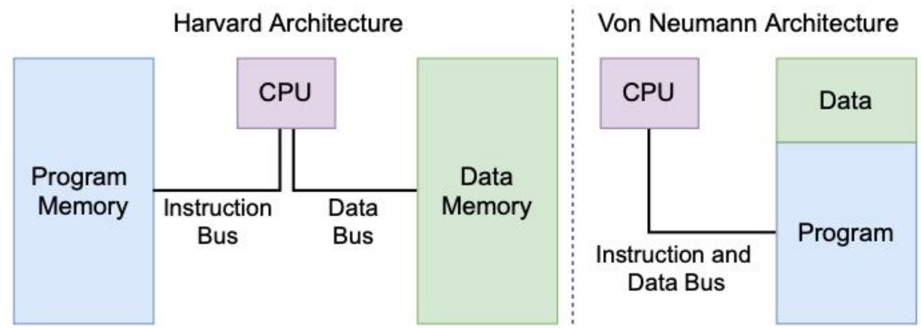
Verified properties

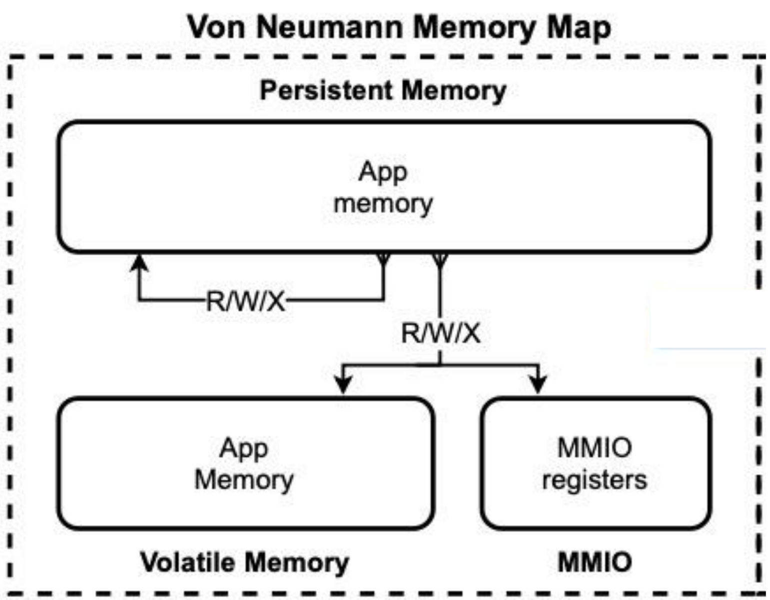
- **Memory-safety: SW is free from the following runtime errors:**
  - Division by zero.
  - Integer overflow / underflow.
  - Buffer overflow / underflow.
  - Out-of-bounding array indexing.
  - Invalid pointer dereferences.
  - Illegal memory accesses.
  - Use after free.
  - Double free.
  - Problematic bit shifts.
  - Type conversions that would overflow the destination.
  - Memory leaks.

- **Freedom from crashes: crash-free is guaranteed at two levels.**
  1. Absence of run-time errors ensures absence of crashes.
     - No segmentation faults (e.g. attempting to write read-only memory).
     - No exceptions (e.g. division by zero).
  2. Atomicity.
     - No failures through scheduled interrupts.

State of the Art






Von Neumann Memory Map
Persistent Memory
App
memory
R/W/X
R/W/X
App
Memory
MMIO
registers
Volatile Memory
MMIO

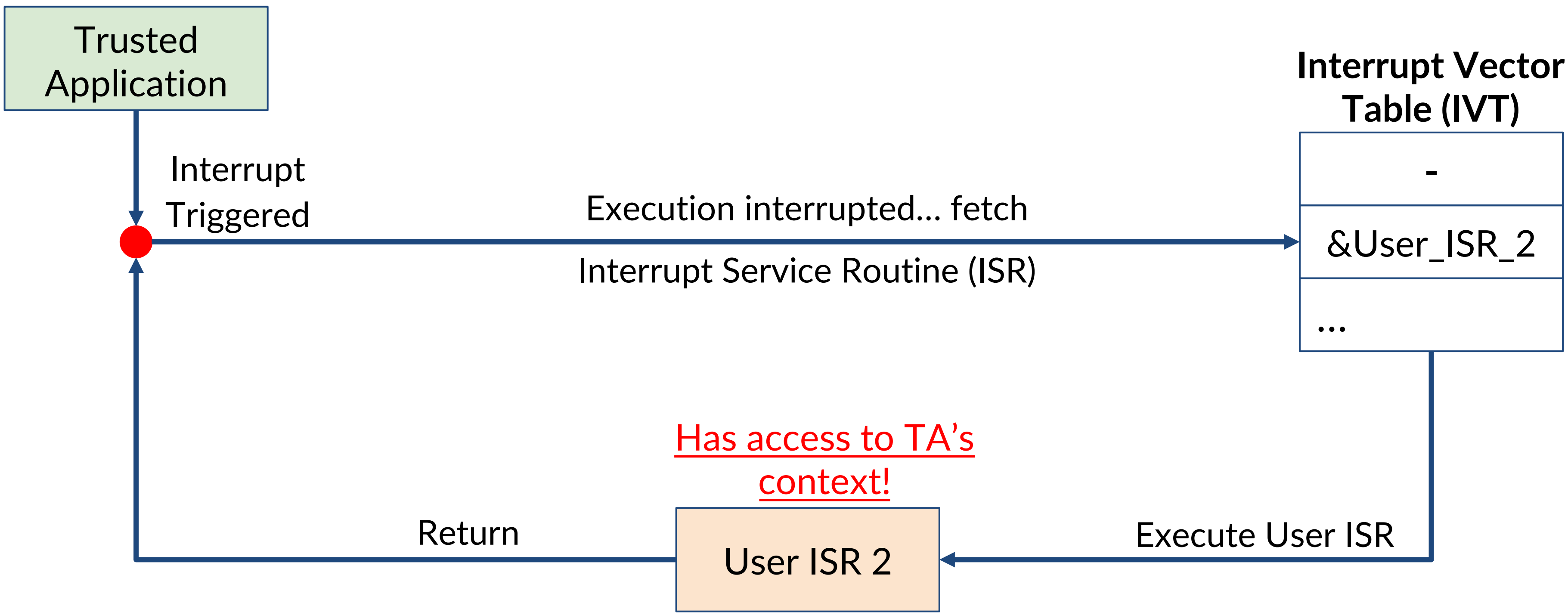
Virtual instructions

```
...
CALL R10 //Dynamic call to an address in a register
```

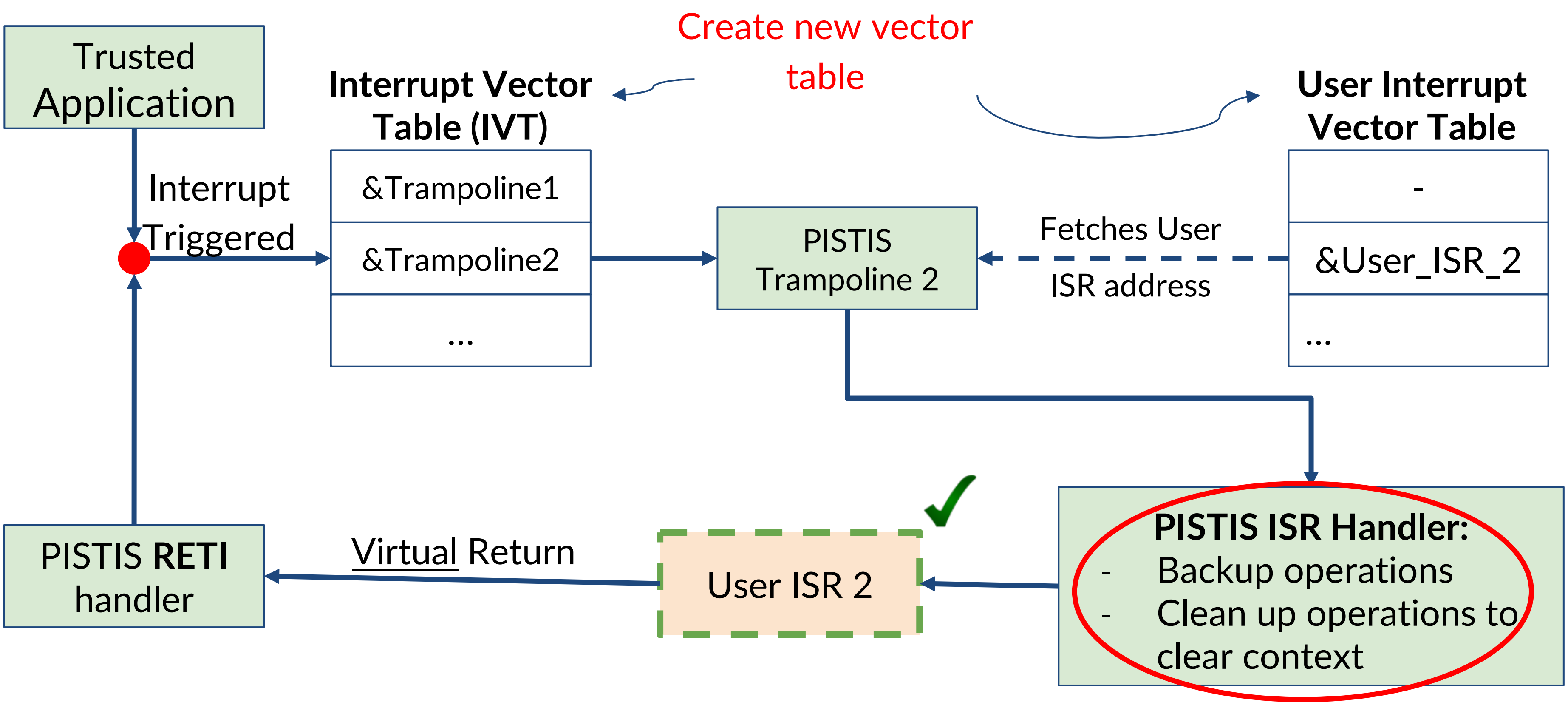
```
...
DINT // Disable interrupts to ensure atomicity
MOV R10, R6 // copy target address to R6
CALL #safe_call // Call to a TCM's safe virt. routine
...
```

```
    ...
safe_call:
    CMP #topInstrMem, R6 // Check upper boundary
    JHS .stopExecution // MCU reset if AP is violated
    CMP #btmInstrMem, R6 // Check bottom boundary
    JL .stopExecution // MCU reset if AP is violated
    EINT //Enable interrupts after passing all checks.
    BR R6 // Jump to original destination
```

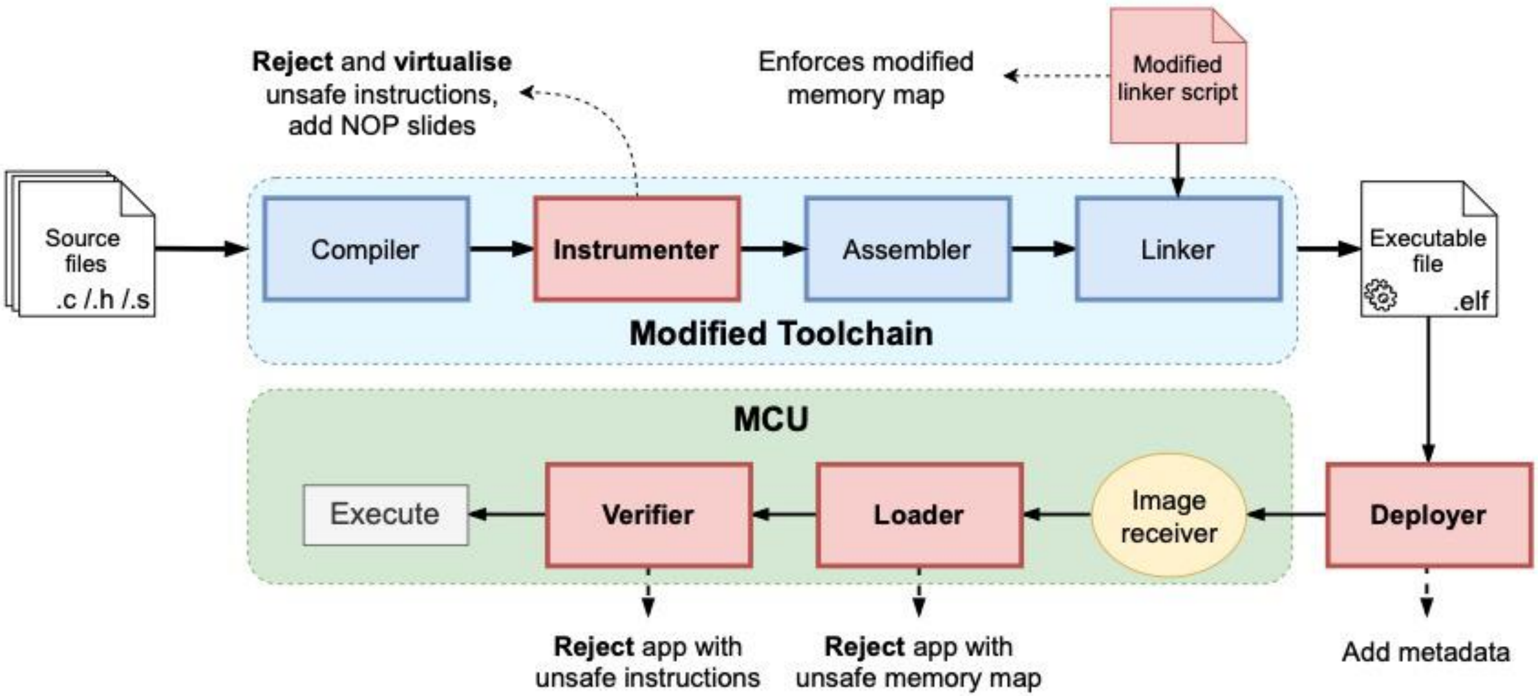
Interrupts


Trusted Application
Interrupt Triggered
Execution interrupted… fetch
Interrupt Service Routine (ISR)
Interrupt Vector Table (IVT)
-
&User_ISR_2
…
Has access to TA's context!
Return
User ISR 2
Execute User ISR

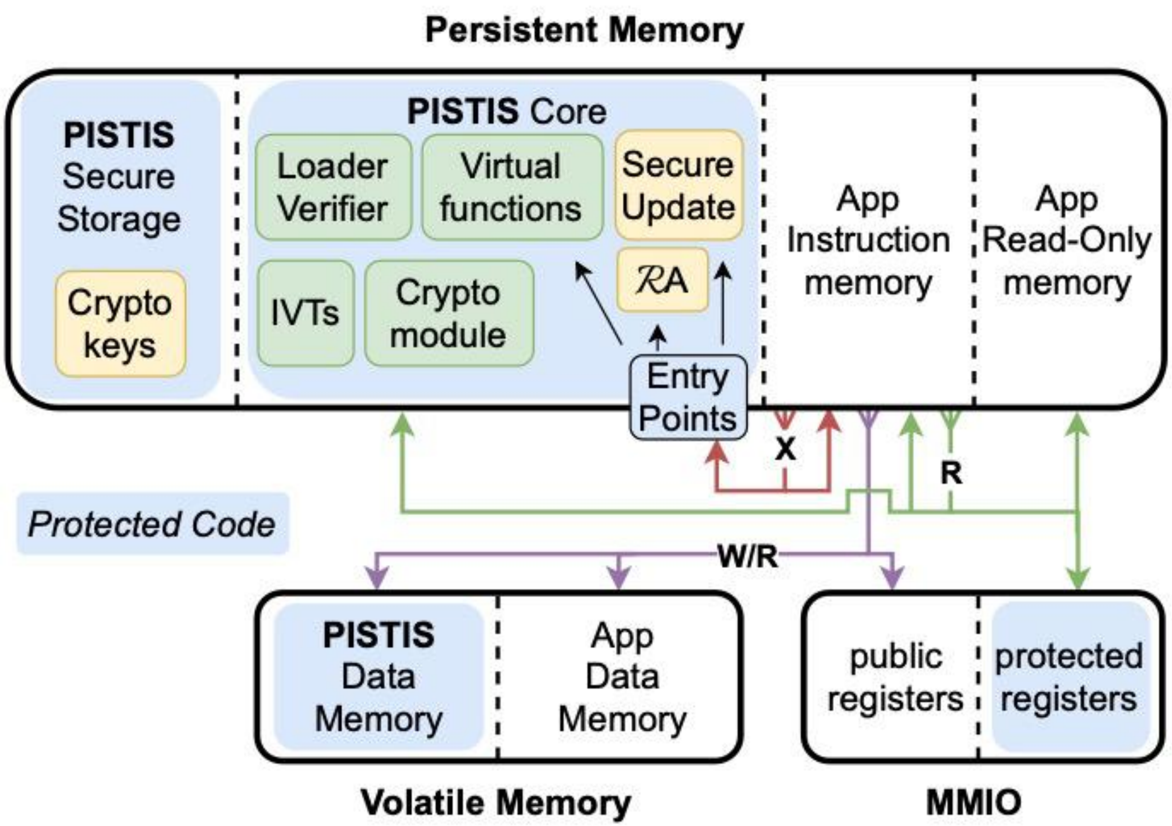
Interrupts


Create new vector table
Trusted Application
Interrupt Vector Table (IVT)
User Interrupt Vector Table
Interrupt Triggered
&Trampoline1
&Trampoline2
...
PISTIS Trampoline 2
Fetches User ISR address
-
&User_ISR_2
...
PISTIS ISR Handler:
- Backup operations
- Clean up operations to clear context
User ISR 2
Virtual Return
PISTIS RETI handler

(Untrusted) Toolchain



PISTIS



Crypto: HCL* library. Formally verified to be memory-safe, functionally-correct and secret-independent

Evaluation

~500 LoC

Overheads of SμV-Enabled Binary Image Sizes

Application	Without SμV	With SμV
Crypto	1414 B	1428 B (**+0.99%**)
Crypto ptr	1438 B	1458 B (**+1.39%**)
Sense temp	1012 B	1034 B (**+2.17%**)
Storage R/W	640 B	652 B (**+1.88%**)
Avg overhead		**1.61%**

Overheads of SμV on the Execution Times of the Sample Applications

Application	Without SμV	With SμV	Relative overhead
Crypto	3.9460 ms	4.1695 ms	5.66%
Crypto ptr	3.9469 ms	4.1706 ms	5.67%
Sense temp	65.9048 ms	65.9364 ms	0.05%
Storage Write	859.6278 ms	859.6897 ms	0.01%
Storage Read	0.4382 ms	0.4468 ms	1.96%
Avg overhead			**2.67%**

8-bit AVR ATmega 1284p MCU running at 10 MHz, with 16 KB of SRAM and 128 KB of flash. Modified Harward Arch.

App	ELF Binary		Memory Footprint	
	Orig.	**Mod.**	**Orig.**	**Mod.**
SerialMSP	3884 B	412 B (**-89.39%**)	302 B	356 B (**+17.88%**)
CopyDMA	5764 B	694 B (**-87.96%**)	444 B	628 B (**+41.44%**)
XorCypher	5940 B	532 B (**-91.04%**)	247 B	475 B (**+92.31%**)
Bitcount	5664 B	1602 B (**-71.72%**)	3684 B	5462 B (**+48.26%**)
SHA-256	9448 B	5518 B (**-41.60%**)	1376 B	1546 B (**+12.35%**)
ML-acc	16616 B	9512 B (**-42.75%**)	6174 B	9452 B (**+53.09%**)
PrimeFactor	33200 B	3650 B (**-89.01%**)	2192 B	3286 B (**+49.91%**)
32bitMath	6036 B	822 B (**-86.38%**)	522 B	766 B (**+46.74%**)
16bitSwitch	3940 B	182 B (**-95.38%**)	102 B	126 B (**+23.53%**)
8bitMatrix	4640 B	916 B (**-80.26%**)	844 B	860 B (**+1.90%**)
MatrixMul	4324 B	572 B (**-86.77%**)	500 B	516 B (**+3.20%**)
firFilter	24912 B	5486 B (**-77.98%**)	3312 B	5430 B (**+63.95%**)
dhrystone	7840 B	2468 B (**-68.52%**)	1335 B	2411 B (**+80.60%**)
Average		**-77.60%**		**+41.17%**

App	**Normal Execution (Orig.)**	**PISTIS-enabled Execution (Mod.)**
SerialMSP	334.1976 ms	335.325 ms (**+0.34%**)
CopyDMA	118.4960 ms	238.656 ms (**+101.40%**)
XorCypher	245.6500 ms	446.104 ms (**+81.60%**)
Bitcount	5.7520 ms	5.786 ms (**+0.59%**)
SHA-256	49.1888 ms	89.046 ms (**+81.03%**)
ML-acc	1456.9092 ms	3311.829 ms (**+127.32%**)
PrimeFactor	4.0810 ms	5.938 ms (**+45.50%**)
32bitMath	0.9310 ms	1.294 ms (**+38.99%**)
16bitSwitch	0.0050 ms	0.006 ms (**+20.00%**)
8bitMatrix	0.5760 ms	0.577 ms (**+0.17%**)
MatrixMul	0.3430 ms	0.344 ms (**+0.29%**)
firFilter	1093.5059 ms	2359.619 ms (**+115.78%**)
dhrystone	102.9200 ms	177.336 ms (**+72.30%**)
Average		**+52.72%**

~1300 LoC

MSP430 MCU, which features ~132 kB of FLASH, ~8kB of SRAM, and up to an 8 MHz of CPU

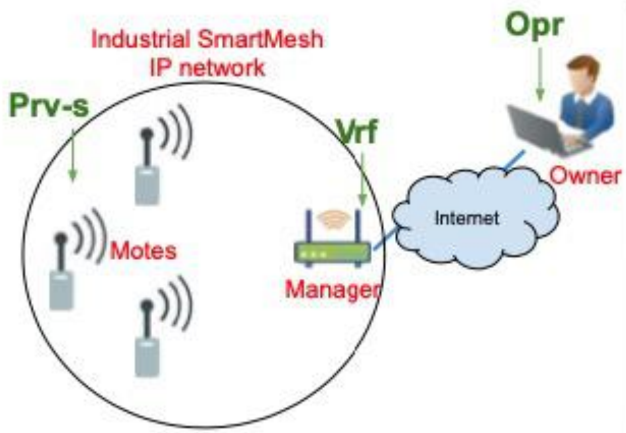
Trusted Applications

- Secure Update
- Shadow stack and CFI
- Verify & Revive

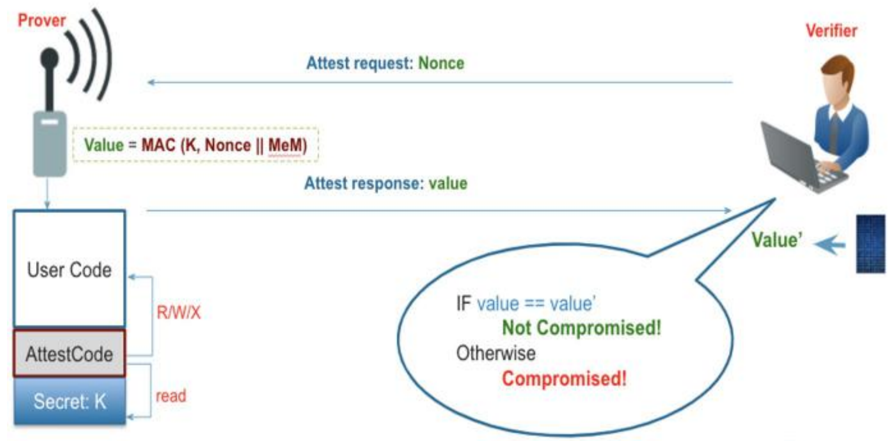
Verify & Revive

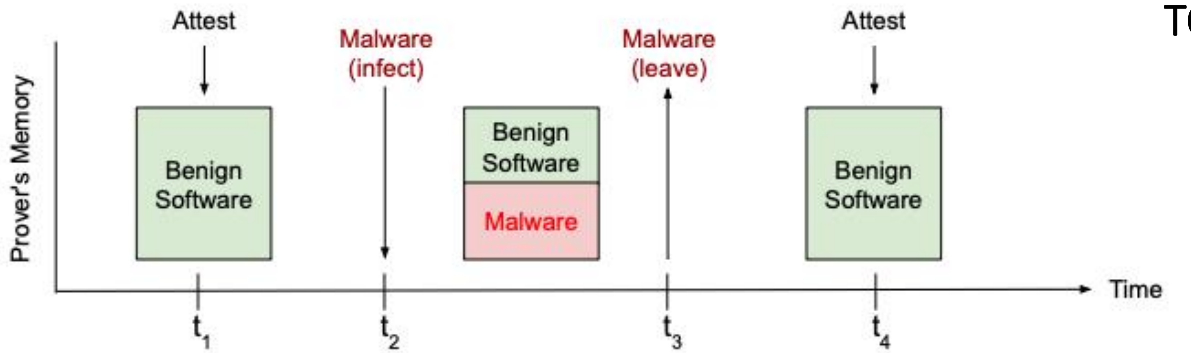
- Composed of
  - RA that mitigate TOCTOU
  - Secure Erasure
  - Remote secure code update for healing

Verify & Revive



(A) A possible scenario of industrial network





TOCTOU problem

VERIFY

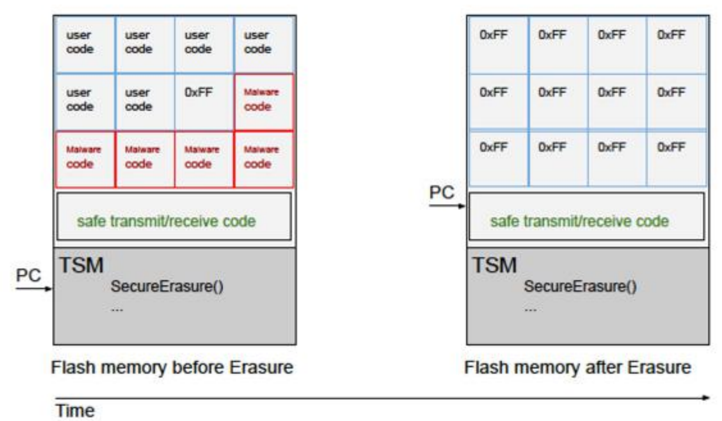
Verifier (Vrf)
K_{auth}, C_V, C_T

Prover (Prv)
K_{auth}, K_{attest}, MAC_{true}, C_p, C_T

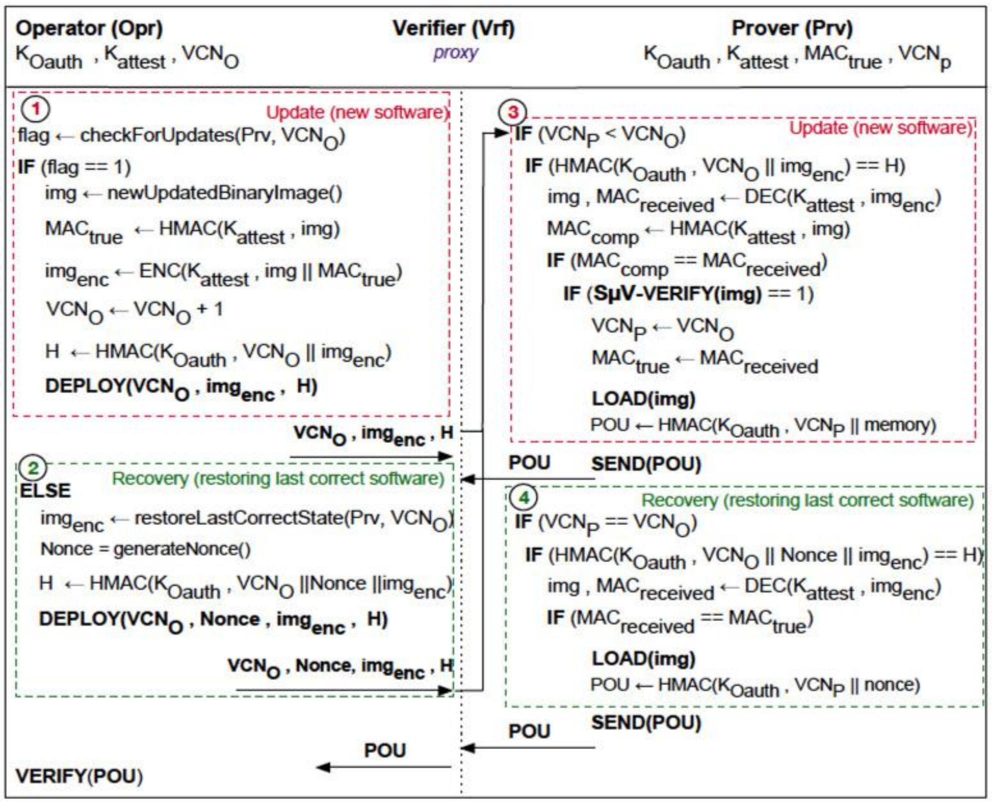
$C_V = C_V + 1$
Nonce = generateNonce()
$H \leftarrow HMAC(K_{auth}$, "attest" || C_V || Nonce)
Attest$_{req}$ = {"attest" || C_V || Nonce }
SEND(Attest$_{req}$, H) → **Attest$_{req}$, H** →

IF ($C_p < C_V$)
  IF (HMAC(K_{auth}, **Attest$_{req}$**) == H)
    $C_p = C_V$
    MAC_{comp} = HMAC(K_{attest}, memory)
    IF (MAC_{comp} == MAC_{true})
      Attest$_{rep}$ = {1 || C_T || HMAC(K_{auth}, 1 || C_T || C_p || Nonce)}
← **Attest$_{rep}$** **SEND(report)**
    ELSE
      Attest$_{rep}$ = {0 || HMAC(K_{auth}, 0 || C_p || Nonce)}
← **Attest$_{rep}$** **SEND(report)**
      perfromSecureErasure()
Verify(Attest$_{rep}$) ← **POE** **sendProofOfSecureErasure()**
checkForProofOfSecureErasureIfRequired()

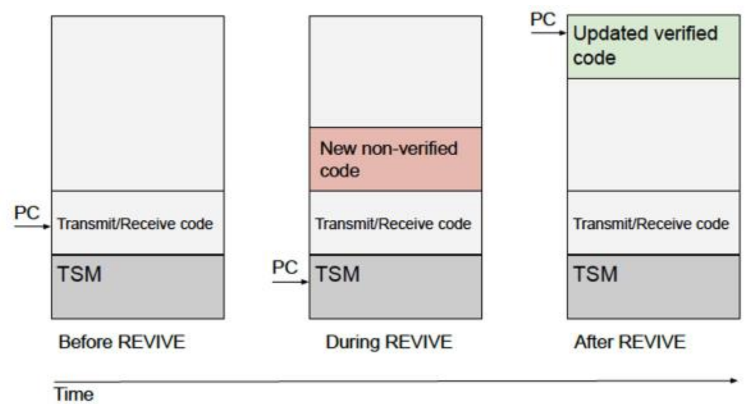
Prv parameters	
K_{auth}	A secret key shared with Vrf for authentication.
K_{attest}	A secret key used for attestation, i.e. computing MAC over entire memory.
C_p	A counter used to avoid replay attacks.
C_T	A counter used to detect TOCTOU attack.
MAC_{true}	A digest computed over a benign state of Prv's memory, using K_{attest}.
K_{Oauth}	A secret key shared with Opr for authentication.
VCN_P	A Version Control Number of current Prv 's software, shared with Opr.
Vrf parameters	
K_{auth}	A secret key shared with Prv for authentication.
C_v	A counter used to avoid replay attacks, initialized with same value like C_P.
C_T	A counter used to detect TOCTOU attack, initialized with same value as Prv's C_T.



REVIVE

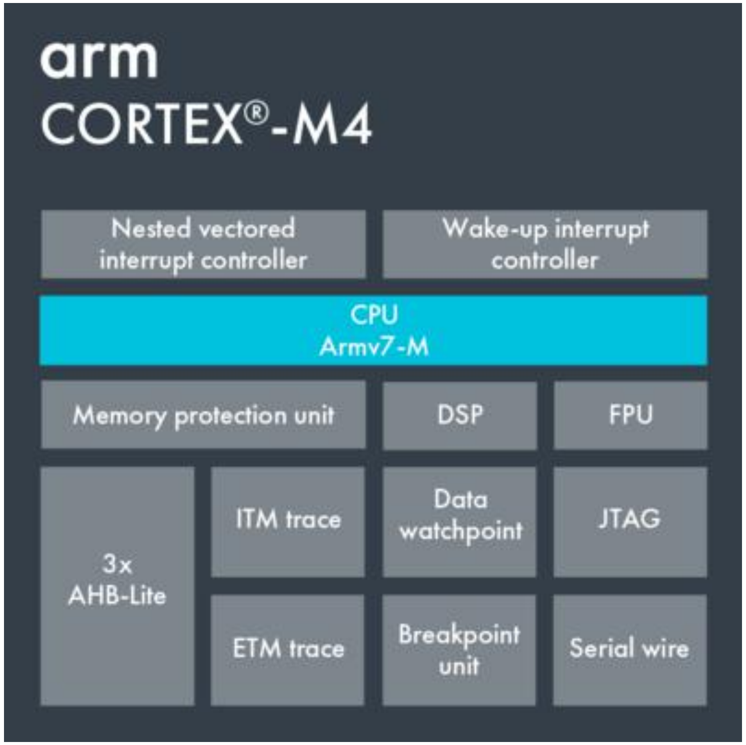


Prv parameters	
K_{auth}	A secret key shared with Vrf for authentication.
K_{attest}	A secret key used for attestation, i.e. computing MAC over entire memory.
C_p	A counter used to avoid replay attacks.
C_T	A counter used to detect TOCTOU attack.
MAC_{true}	A digest computed over a benign state of Prv's memory, using K_{attest}.
K_{Oauth}	A secret key shared with Opr for authentication.
VCN_P	A Version Control Number of current Prv 's software, shared with Opr.
Vrf parameters	
K_{auth}	A secret key shared with Prv for authentication.
C_v	A counter used to avoid replay attacks, initialized with same value like C_p.
C_T	A counter used to detect TOCTOU attack, initialized with same value as Prv's C_T.



Considerations

Hardware must be trusted



Not always clear for what though

A vulnerability/error in HW can be very costly

- i.e., Xilinx 7-s [Usenix Sec 20]

Do it right is getting difficult

- i.e., Intel MPX

Alternatively, HW can be reconfigured/fixed but at the price of loosing some trust

Interoperability issues

Conclusions

- New foundational TAs besided the existing ones
- Now a good time also to re-think about trusted computing architectures. RISC V is an opportunity
- We didn’t touch side-channels attacks

References

M. Salehi, L. Degani, M. Roveri, D. Hughes, B. Crispo, Discovery and Identification of Memory Corruption Vulnerabilities on Bare-Metal Embedded Devices. IEEE Trans. Dependable and Secure Computing (2023)

M. Grisafi, M. Ammar, M. Roveri, B. Crispo, PISTIS: Trusted Computing Architecture for Low-end Embedded Systems. USENIX Security Symposium (2022)

M. Salehi, G. De Borger, D. Hughes, B. Crispo, NemesisGuard: Mitigating interrupt latency side channel attacks with static binary rewriting. Computer Networks (2022)

M. Grisafi, M. Ammar, K. Sinan Yildirim, B. Crispo, MPI: Memory Protection for Intermittent Computing. IEEE Trans. on Information Forensics and Security (2022)

M. Ammar, B. Crispo, Verify&Revive: Secure Detection and Recovery of Compromised Low-end Embedded Devices. Annual Computer Security Applications Conference (ACSAC 2020)

M. Ammar, B. Crispo, G. Tsudik, SIMPLE: A Remote Attestation Approach for Resource-constrained IoT devices. ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS 2020)

M. Salehi, D. Hughes, B.Crispo, µSBS: Static Binary Sanitization of Bare-metal Embedded Devices for Fault Observability. International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)

M. Ammar, B. Crispo, B. Jacobs, D. Hughes, W. Daniels, SµV - The Security MicroVisor: A Formally-verified Software-based Security Architecture for the Internet of Things. IEEE Trans. Dependable and Secure Computing (2019)