



## Leveraging Intel DCI for Memory Forensics

By:

Tobias Latzo (Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)), Matti Schulze (FAU), and Felix Freiling (FAU)

*From the proceedings of*

The Digital Forensic Research Conference

**DFRWS USA 2021**

July 12-15, 2021

DFRWS is dedicated to the sharing of knowledge and ideas about digital forensics research. Ever since it organized the first open workshop devoted to digital forensics in 2001, DFRWS continues to bring academics and practitioners together in an informal environment.

As a non-profit, volunteer organization, DFRWS sponsors technical working groups, annual conferences and challenges to help drive the direction of research and development.

**<https://dfrws.org>**

# Leveraging Intel DCI for Memory Forensics

---



Tobias Latzo, Matti Schulze, and Felix Freiling

July 12, 2021

Security Research Group

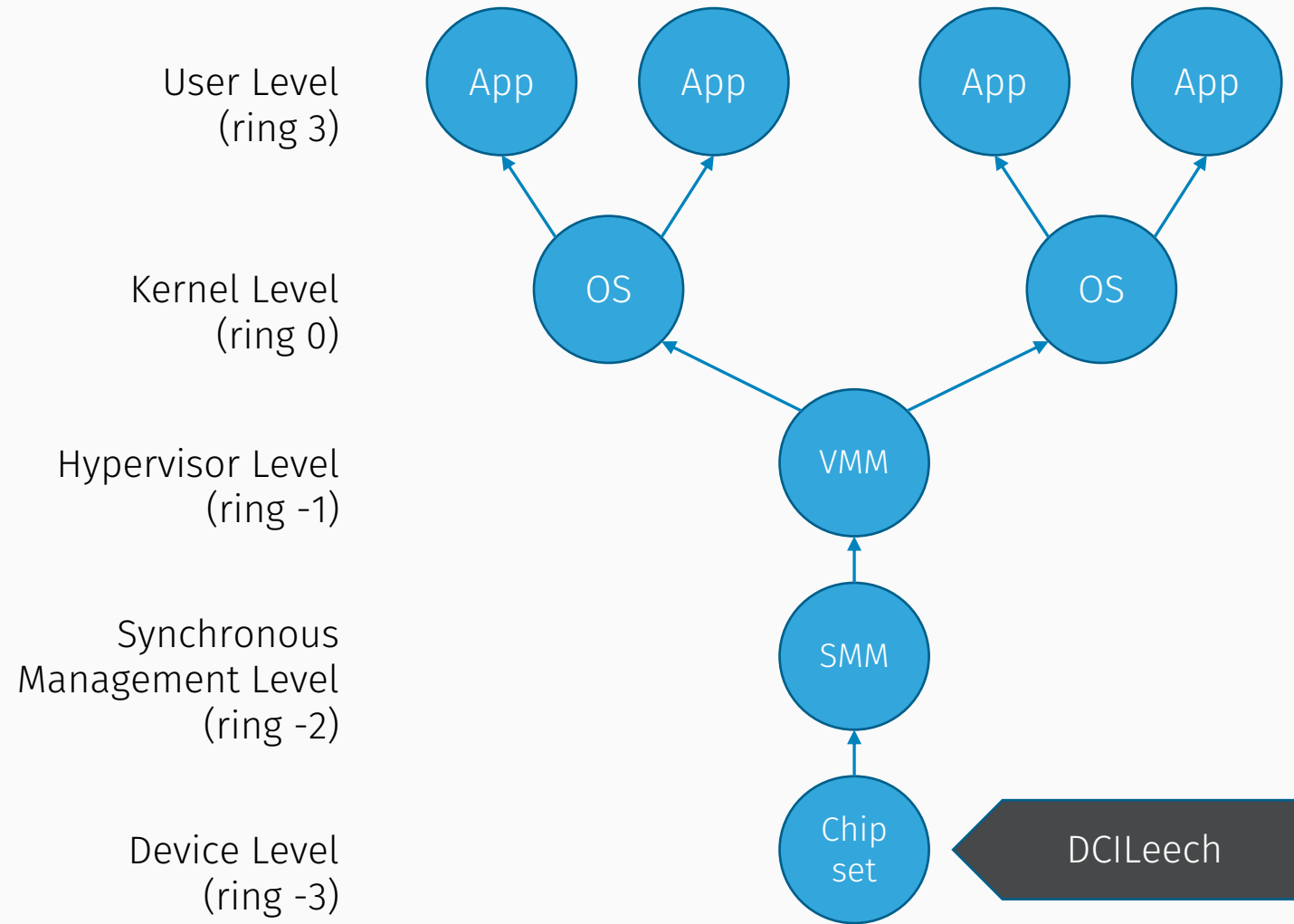
Department of Computer Science

Friedrich-Alexander University Erlangen-Nürnberg (FAU)

# Increasing importance of memory forensics

- Running processes
- Encryption keys
- Decrypted data
- Network attached storage
- Fileless malware

# Memory access hierarchy<sup>1</sup>



<sup>1</sup>T. Latzo, R. Palutke, and F. Freiling. *A universal taxonomy and survey of forensic memory acquisition techniques*. Digital Investigation, 28(Supplement):56-69, 2019.

# Criteria of memory acquisition techniques

Definition by Vömel and Freiling<sup>1</sup>:

- Correctness
  - Actual values when snapshot was taken
- Atomicity
  - No inconsistencies due to interleaving memory acquisition
  - Inconsistencies are frequent and have negative impact on the analysis<sup>2</sup>
- Integrity
  - Content of memory is not changed after investigator decides to take a snapshot

<sup>1</sup> S. Vömel, F. Freiling: Correctness, atomicity, and integrity: Defining criteria for forensically-sound memory acquisition. Digital Investigation. 2012

<sup>2</sup> F. Pagani, Fabio, O. Fedorov, and D. Balzarotti. *Introducing the temporal dimension to memory forensics*. ACM TOPS 22.2, 2019

# Intel Direct Connect Interface (DCI)

- Low-cost closed chassis JTAG debugging via an USB 3 A-to-A cable
- Nearly unlimited access to the hardware
- Can often be enabled using hidden firmware flags
- On target side no software required

DCI is very powerful and can even halt the CPU.  
Let's use it for atomic memory dumps.

# Previous work

- Most work by Goryachy and Ermolov
  - Tapping into the Core (33c3)
  - Inside Intel Management Engine (34c3)
  - Intel DCI Secrets (HITBSecConf 2017)
- Firmware debugging by Jauregui
  - Intro to Closed Chassis Debugging (OSFC 2019)

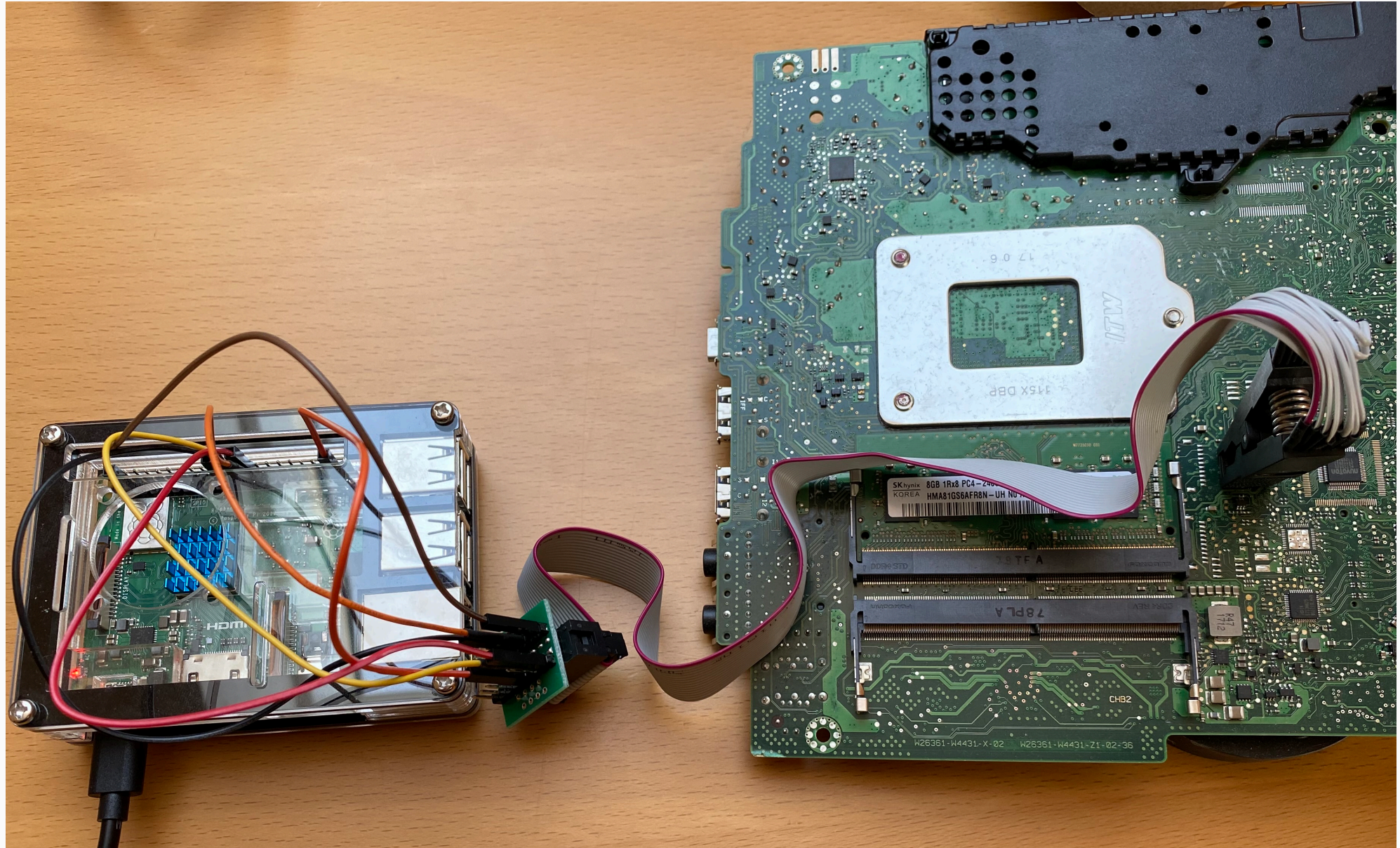
Activation



# Flags to enable Intel DCI

Flag	Value	Description
Debug Interface	1	Enables silicon debug features
Debug Interface Lock	0	Allows changes of the MSR
Direct Connect Interface	1	Enables DCI
DCI Enable (HDCIEN)	1	Indicates DCI is enabled

# Read the EEPROM via SPI





# Modifying the firmware flags

AMIBCP Version 5.02.0023 - [flash.bin]

File View Window Help

Setup Configuration | PCI IRQ Routing | BIOS Strings | DMI Tables | Boot Order | BIOS Features

Advanced

- Serial ATA Controller 0 Settings
- Serial ATA Controller 1 Settings
- Serial ATA Controller 2 Settings
- Global Acoustic Configuration
- SMART Settings
- Serial Port 1 Configuration
- Auto BIOS Update
- Admin
  - System Management
  - Driver Health
  - Chipset
    - System Agent (SA) Configuration
    - PCH-IO Configuration
      - PCI Express Configuration
      - USB Configuration
      - SATA And RST Configuration
      - HD Audio Configuration
      - Security Configuration
      - SerialIo Configuration
      - SkyCam Configuration
      - SCS Configuration
      - ISH Configuration
      - TraceHub Configuration
      - Pch Thermal Throttling Control
    - SB Porting Configuration
    - Extra options
    - Extra options

Handle	Control Group Structures	Show	Access/Use	Failsafe	Optimal
(0C79)	SATA And RST Configuration	Yes	Default		
(0C0C)	USB Configuration	Yes	Default		
(083F)	Security Configuration	Yes	Default		
(07CE)	HD Audio Configuration	Yes	Default		
(0F86)	SerialIo Configuration	Yes	Default		
(0D71)	SkyCam Configuration	Yes	Default		
(0F5F)	SCS Configuration	Yes	Default		
(0F70)	ISH Configuration	Yes	Default		
(101A)	TraceHub Configuration Menu	Yes	Default		
(102E)	Pch Thermal Throttling Control	Yes	Default		
(00B8)	SB Porting Configuration	Yes	Default		
(000F)		Yes	Default		
(07B5)	DCI enable (HDCIEN)	Yes	Default	Disabled	Disabled
(07AD)	USB/UART	Yes	Default	UART	UART
(07B1)	Debug Port Selection	Yes	Default	Legacy UART	Legacy UART
(1010)	GNSS	Yes	Default	Disabled	Disabled
(100F)	GNSS Device Model	Yes	Default	CG2000	CG2000
(07B7)	PCH LAN Controller	Yes	Default	Enabled	Enabled
(0D58)	Sensor Hub Type	Yes	Default	None	None
(0D5D)	DeepSx Power Policies	Yes	Default	Disabled	Disabled
(0D63)	LAN Wake From DeepSx	Yes	Default	Enabled	Enabled
(07BA)	Wake on LAN Enable	Yes	Default	Enabled	Enabled
(07BE)	SLP_LAN# Low on DC Power	Yes	Default	Enabled	Enabled
(07BC)	K1 off	Yes	Default	Disabled	Disabled
(07C0)	Wake on WLAN and BT Enable	Yes	Default	Disabled	Disabled
(07C2)	DeepSx Wake on WLAN and BT...	Yes	Default	Disabled	Disabled
(07C4)	Disable DSX ACPI PRESENT PULD...	Yes	Default	Disabled	Disabled
(0835)	CLKRUN# logic	Yes	Default	Disabled	Disabled
(0D65)	Serial IRQ Mode	Yes	Default	Quiet	Continuous
(0D69)	Port 61h Bit-4 Emulation	Yes	Default	Enabled	Enabled
(07CC)	High Precision Timer	Yes	Default	Enabled	Enabled
(0C66)	State After G3	Yes	Default	S0 State	S0 State

Menu Help String

Control Help String

Select Kernel Debug Port and report in ACPI DBG2 table

LogFile

Undo

Ready

NUM

# DCILeech Implementation

- DMA Attack Framework by Ulf Frisk<sup>1</sup>
- Features:
  - Read and write memory
  - Inject kernel code on the target
  - Push and pull files
  - Shorting Windows login screen
  - Etc.

```
enum rawtcp_cmd {  
    STATUS,  
    MEM_READ,  
    MEM_WRITE  
}
```

```
struct rawtcp_msg {  
    enum_rawtcp_cmd cmd,  
    uint64_t addr,  
    uint64_t size  
}
```

<sup>1</sup> <https://github.com/ufrisk/pcileech>

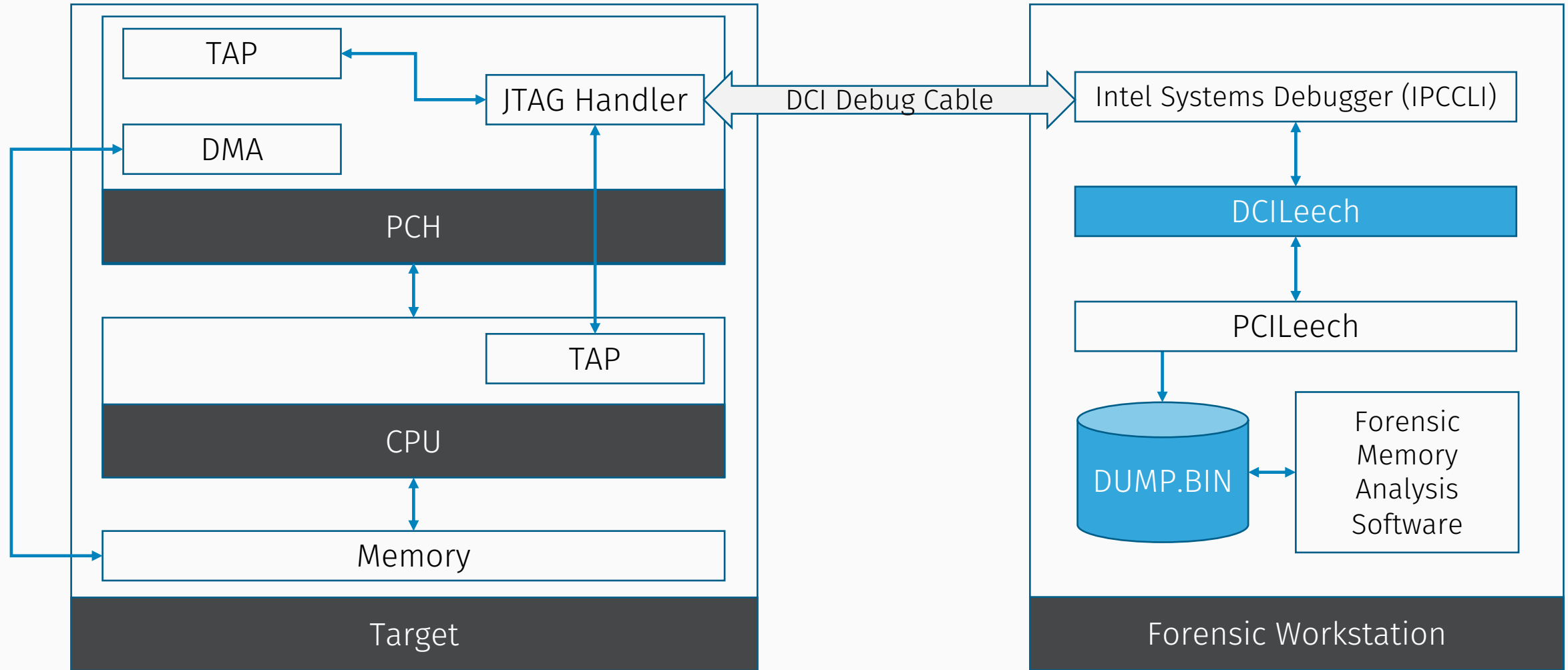
# PCILeech extension

Problem: PCILeech expects injected code to be executed

- For many PCILeech features a kernel module is injected
- PCILeech waits for a physical address that is written by the injected code
- However: CPU is halted
- Solution: Extend commands by **GO** and **HALT**

```
enum rawtcp_cmd {  
    STATUS,  
    MEM_READ,  
    MEM_WRITE,  
    DCI_GO,  
    DCI_HALT  
}
```

# Intel DCI: Architecture

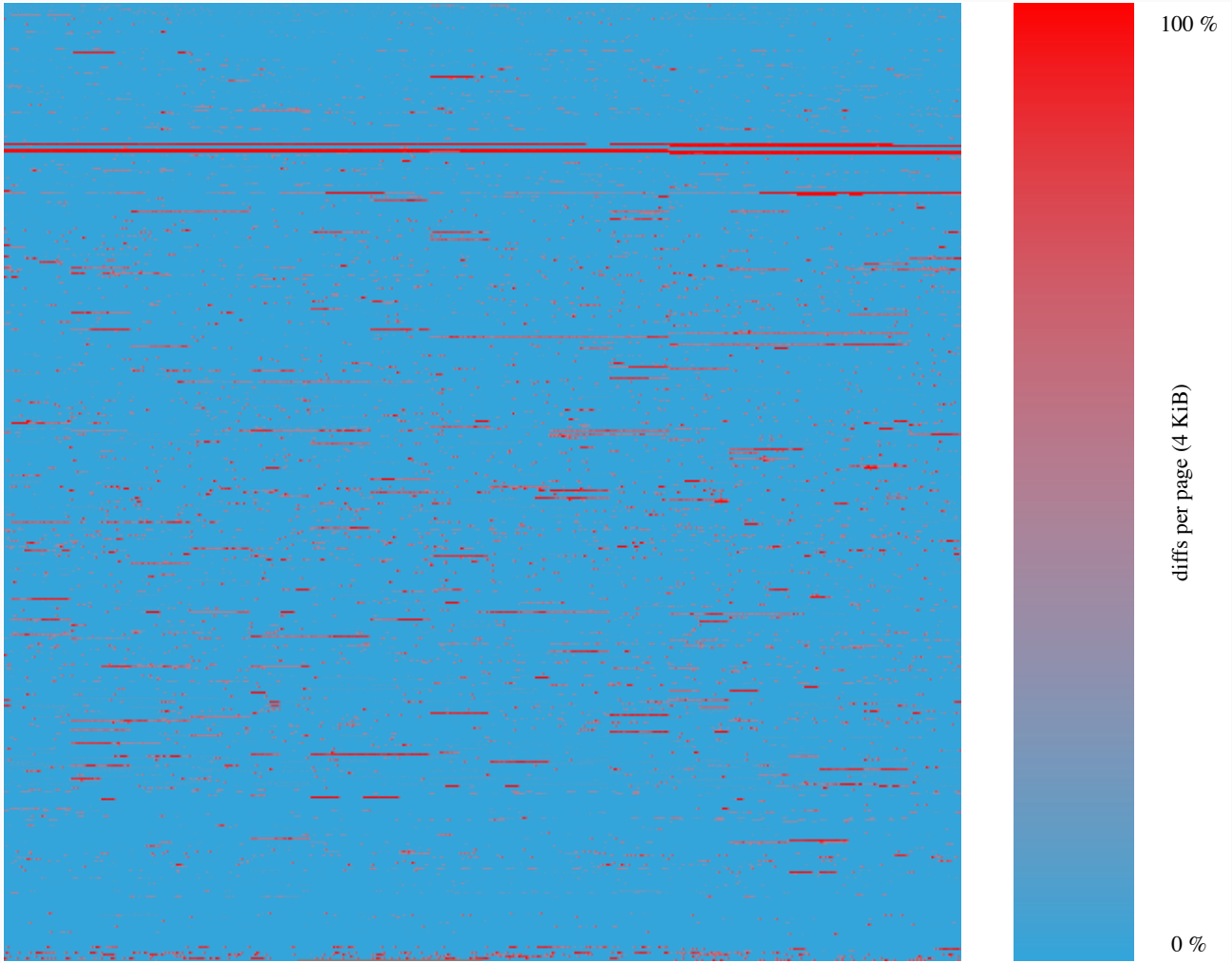


Evaluation



# Correctness

## Comparison with a LiME dump

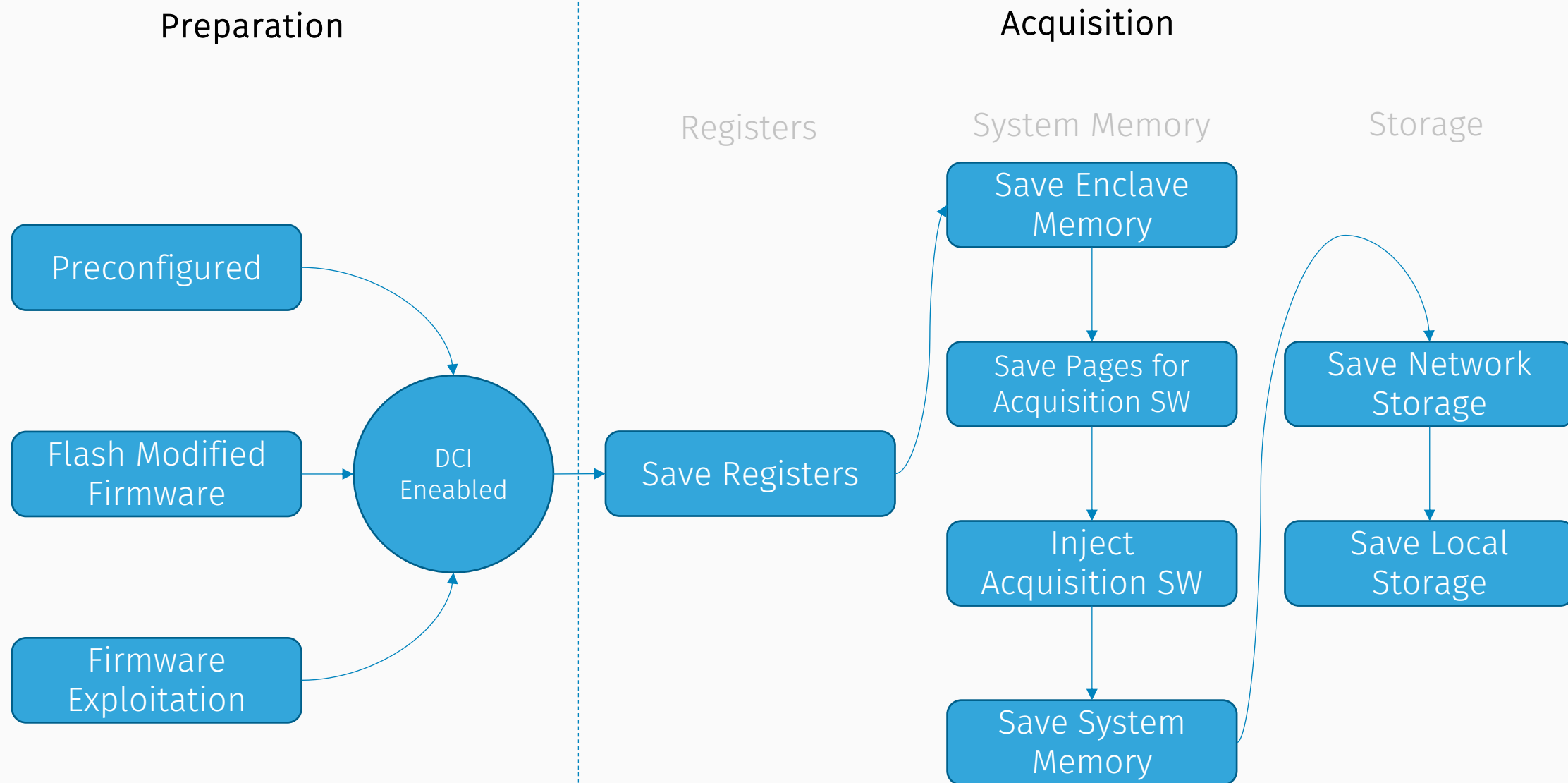


# PCILeech payloads

- Memory snapshot: `dump`
- Kernel module injection: `kmdload`
- File retrieval and pushing: `lx64_filepull` and `lx64_filepush`
- Windows 10 unlock: `wx64_unlock`

- Test program: Load Lena test image into protected enclave memory
- Find address of EPC:
  - `cpuid -l 0x12 -s 0x2`
- Read enclave memory using ITPII: `edbgread`
- Only tested with Debug profile
  - Release profile probably possible with `set_debugoptin`

# Forensic Triage with Intel DCI



# Conclusion

# Conclusion

- Correctness ✓
  - Evaluation showed DCILeech is working properly
- Atomicity: ✓
  - CPU is stopped
- Integrity: ✓
  - No software on target side
- Acquisition of registers possible
- Intel SGX Enclaves can be dumped
  - Debug mode ✓
  - Release mode ?
- Bad performance: 70 KiB/s (SGX 4 KiB/s)
- Limited for on-site investigations → forensic-readiness



Thank you!