POLITECNICO
MILANO 1863

DIPARTIMENTO DI ELETTRONICA
INFORMAZIONE E BIOINGEGNERIA

Packers & Evasion

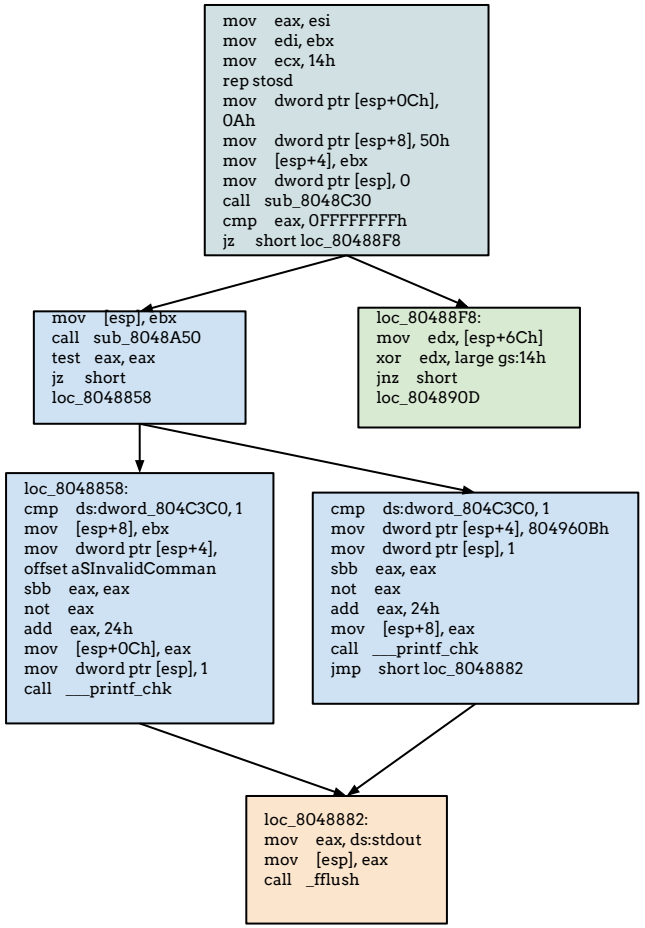
Mario Polino,

Malware Analysis

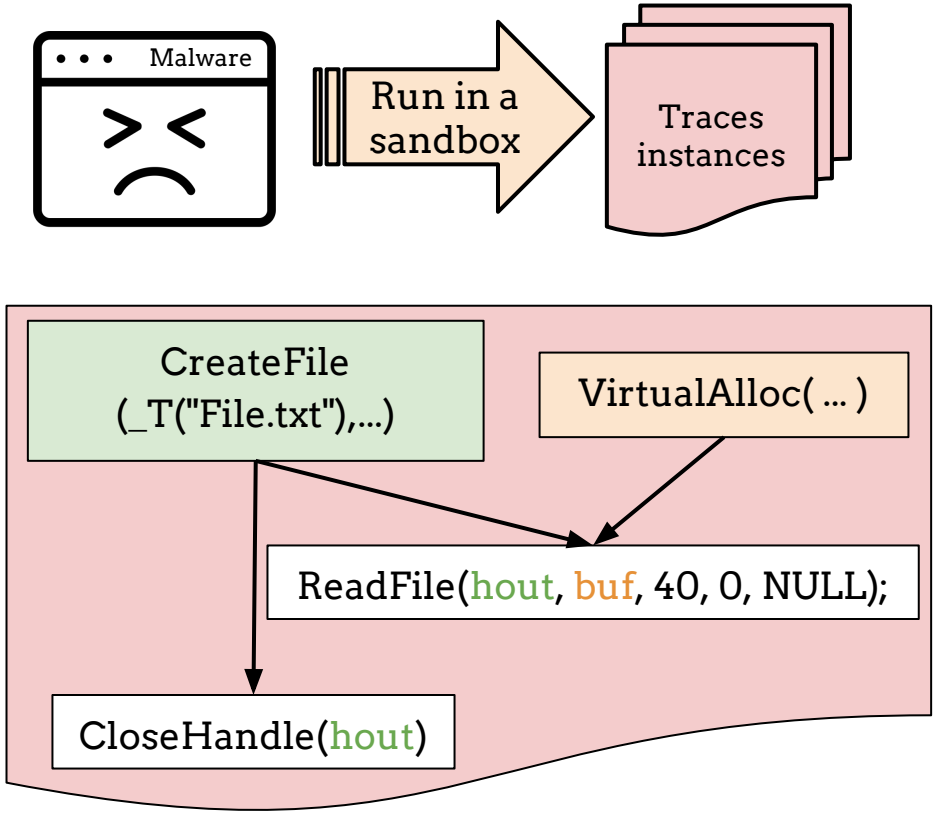
POLITECNICO
MILANO 1863

Malware Analysis

Static



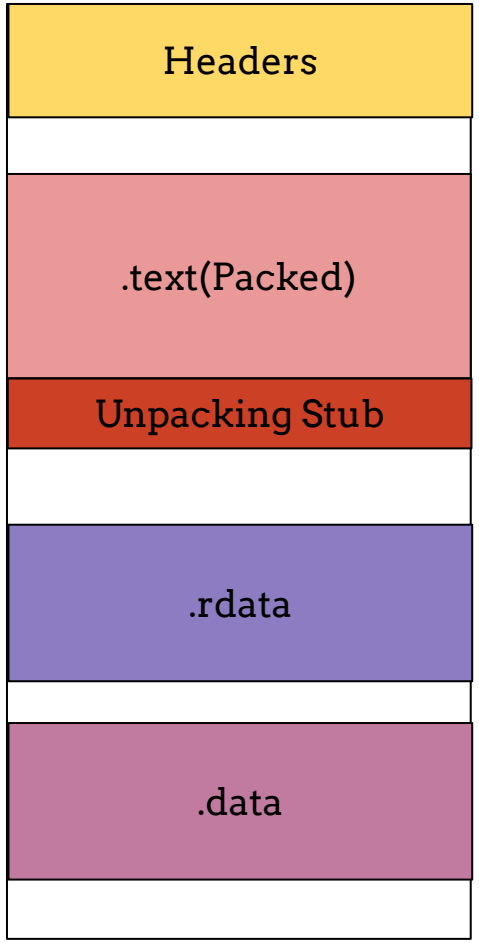
Dynamic



POLITECNICO MILANO 1863

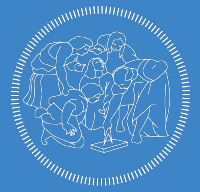
Static & Dynamic Issues

Packers



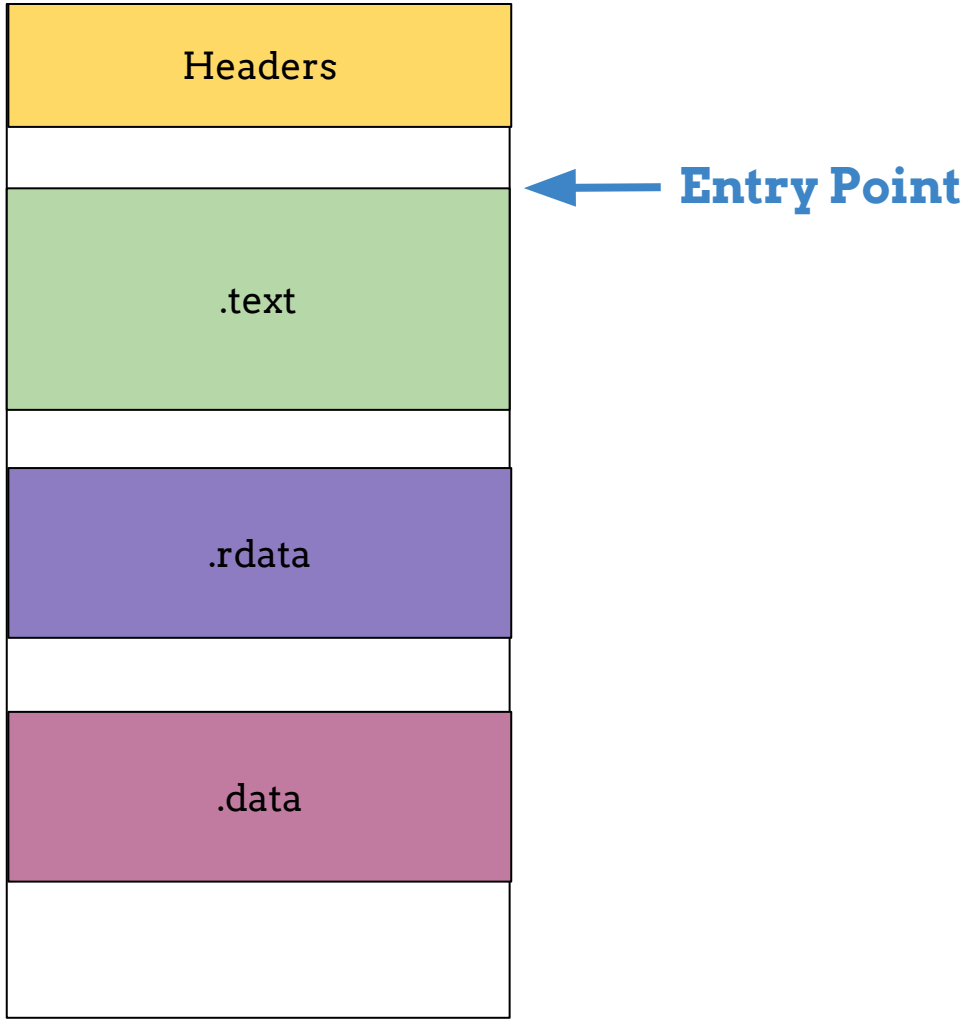
Evasive Malware

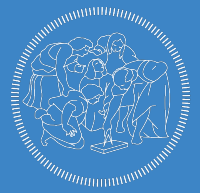
```
If (amIUnderAnalysis())
{
    die();
}
else
{
   beMalicious();
}
```



POLITECNICO
MILANO 1863

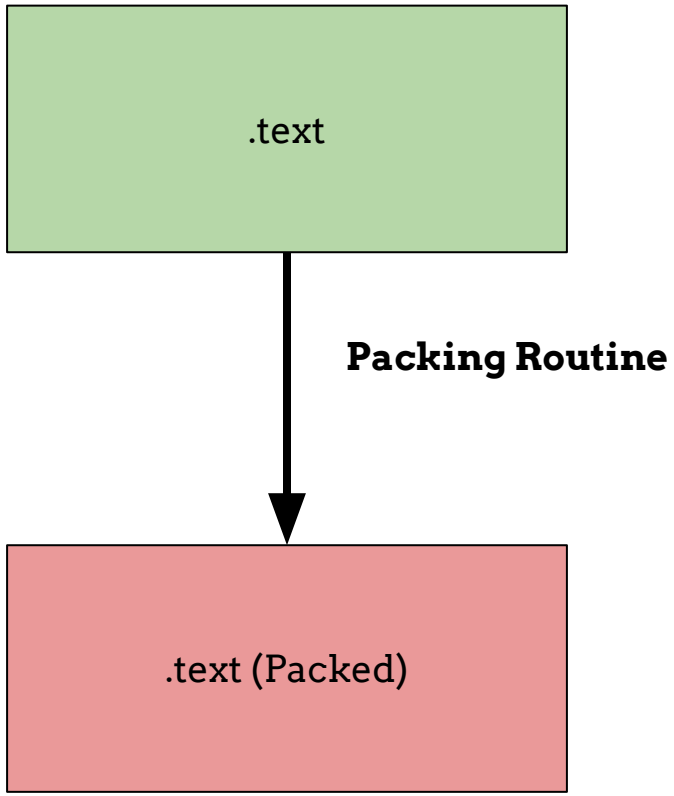
How does a packed malware work?

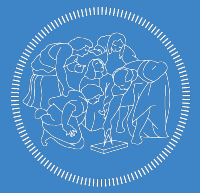




POLITECNICO
MILANO 1863

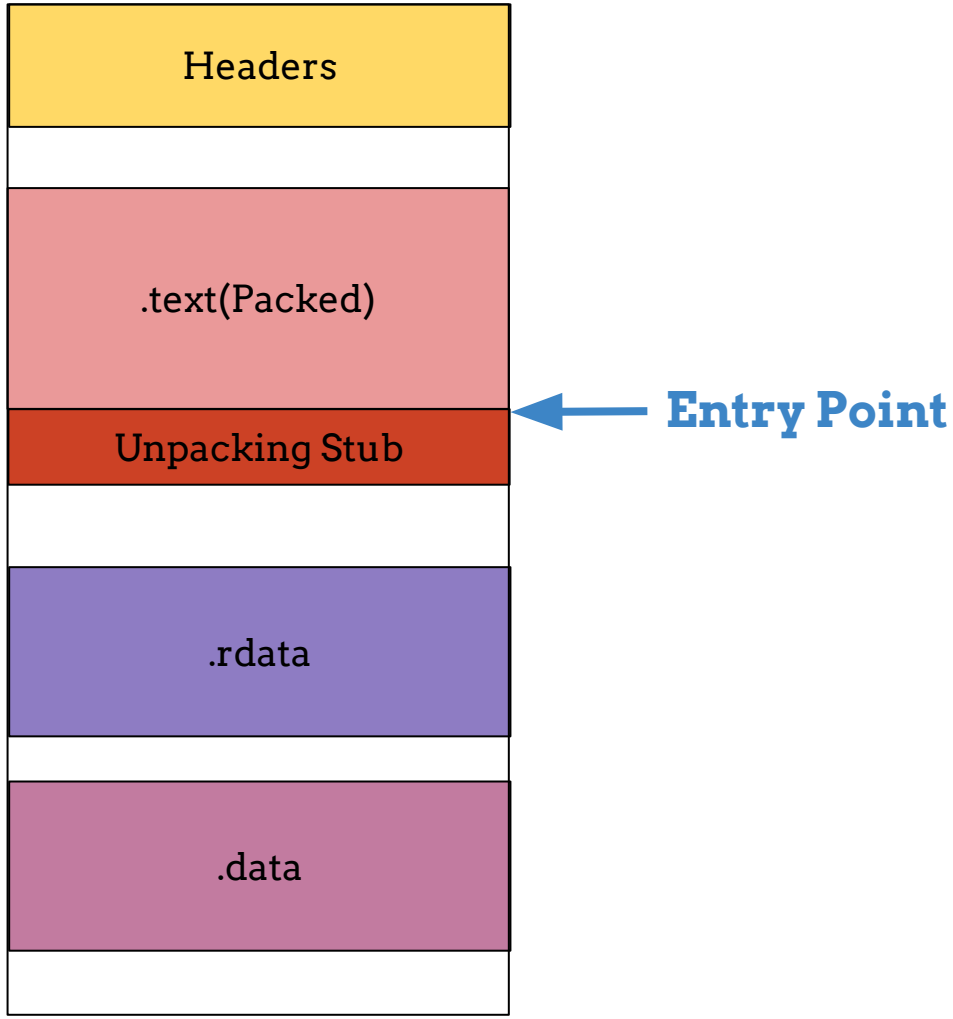
How does a packed malware work?

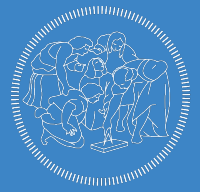




POLITECNICO
MILANO 1863

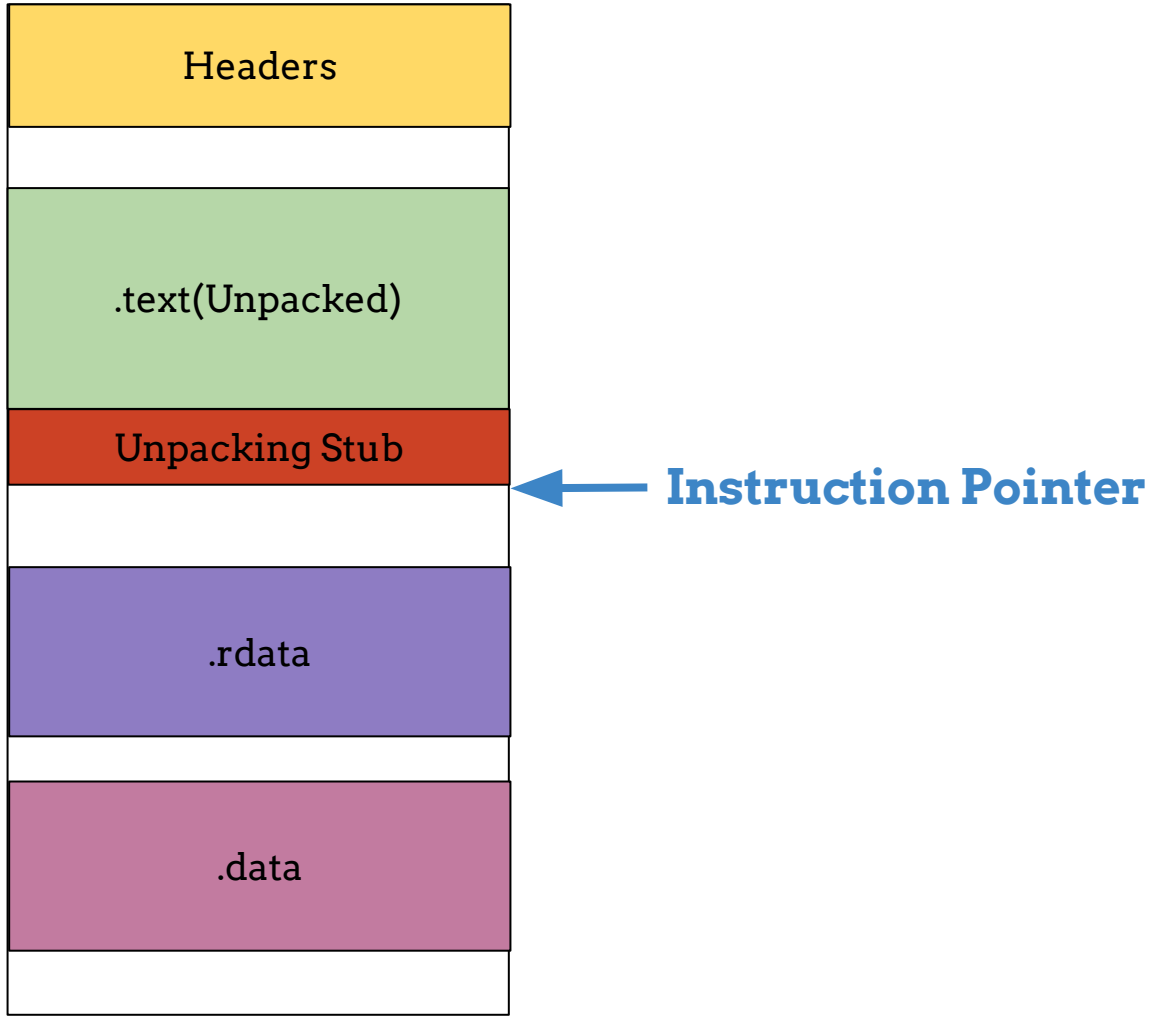
How does a packed malware work?





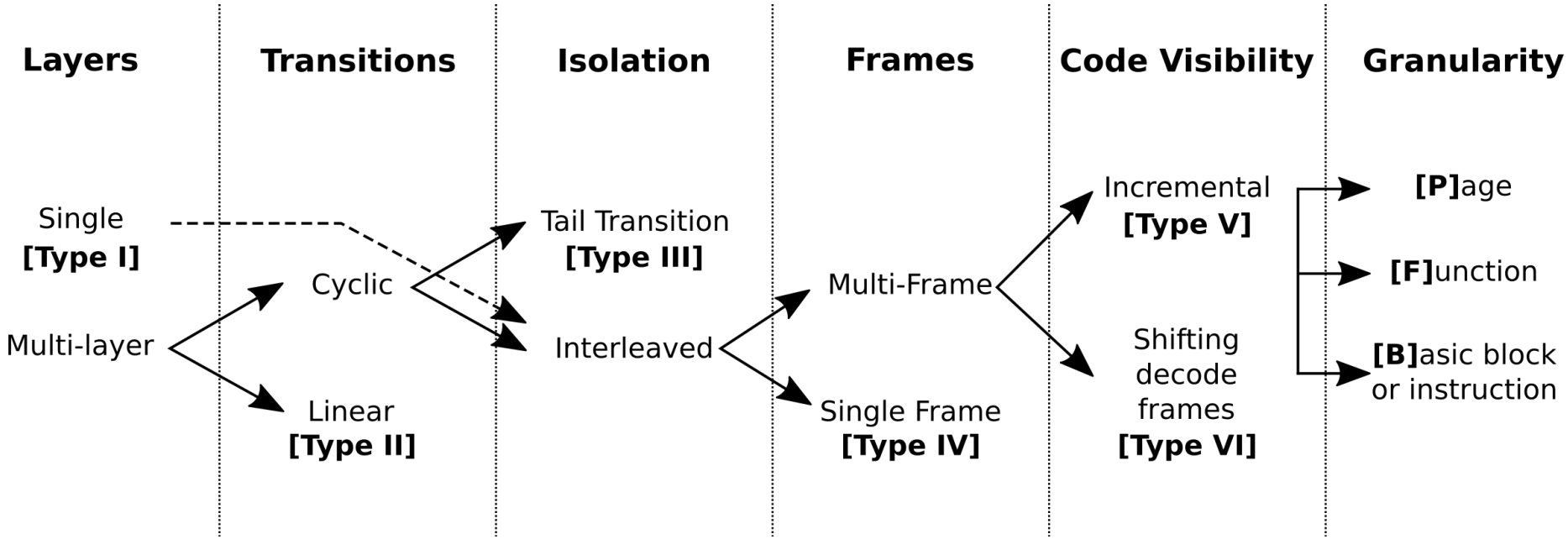
POLITECNICO
MILANO 1863

How does a packed malware work?



Packers Complexity

From **Unpacked** to **Custom VM** there is a huge **gray** area

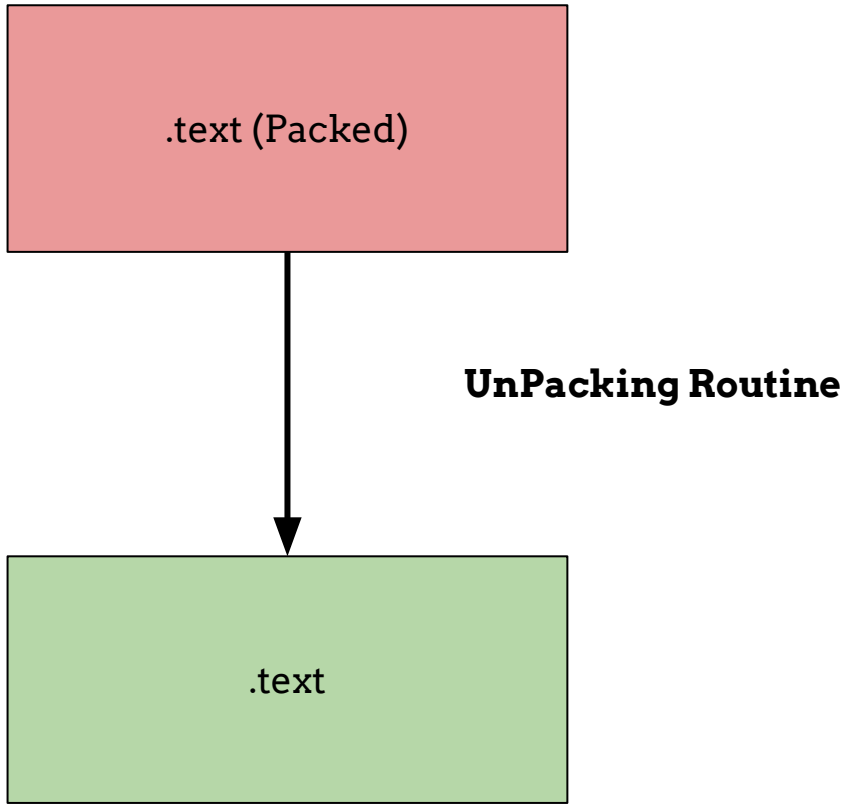


SoK: Deep Packer Inspection: A Longitudinal Study of the Complexity of Run-Time Packers

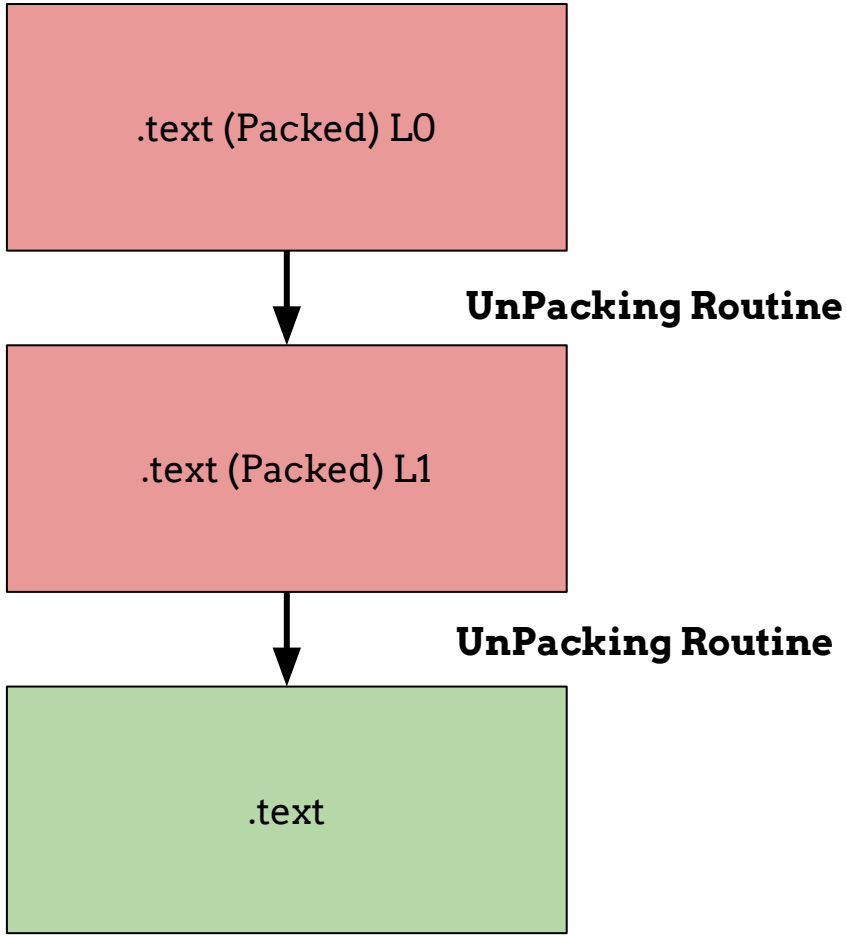
POLITECNICO
MILANO 1863

Layers

Type I



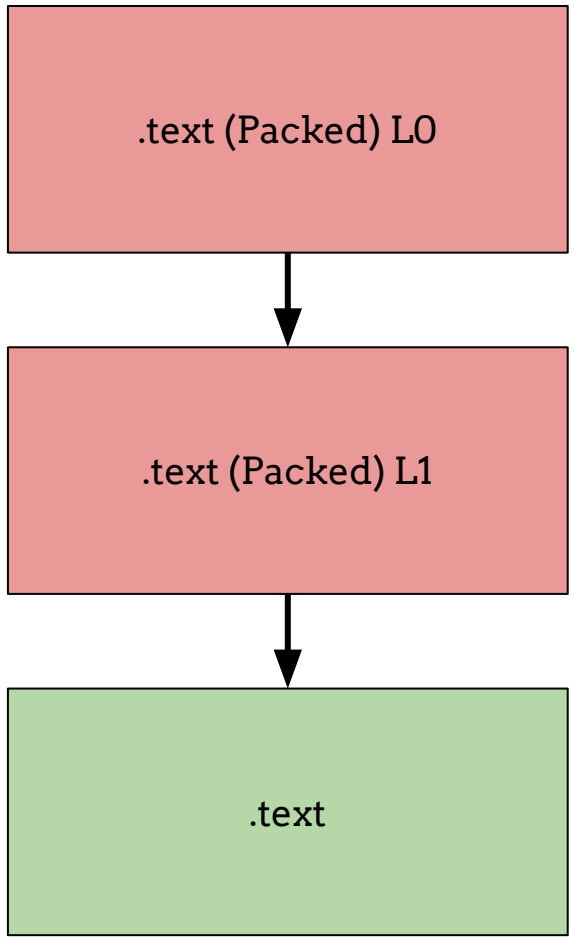
Type >= II



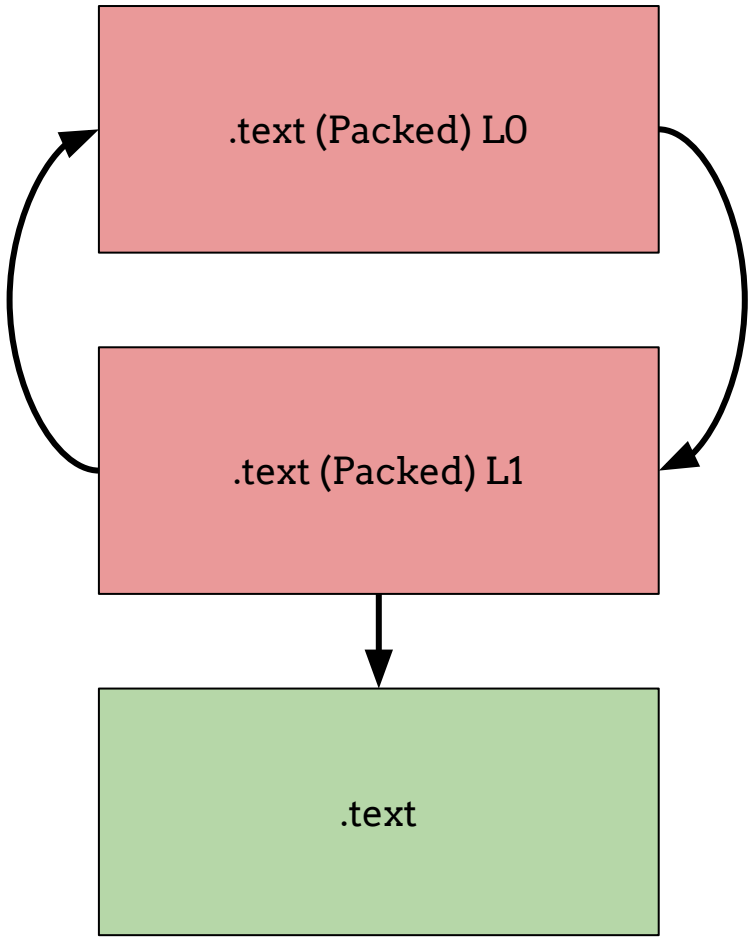
POLITECNICO
MILANO 1863

Transaction & Isolation

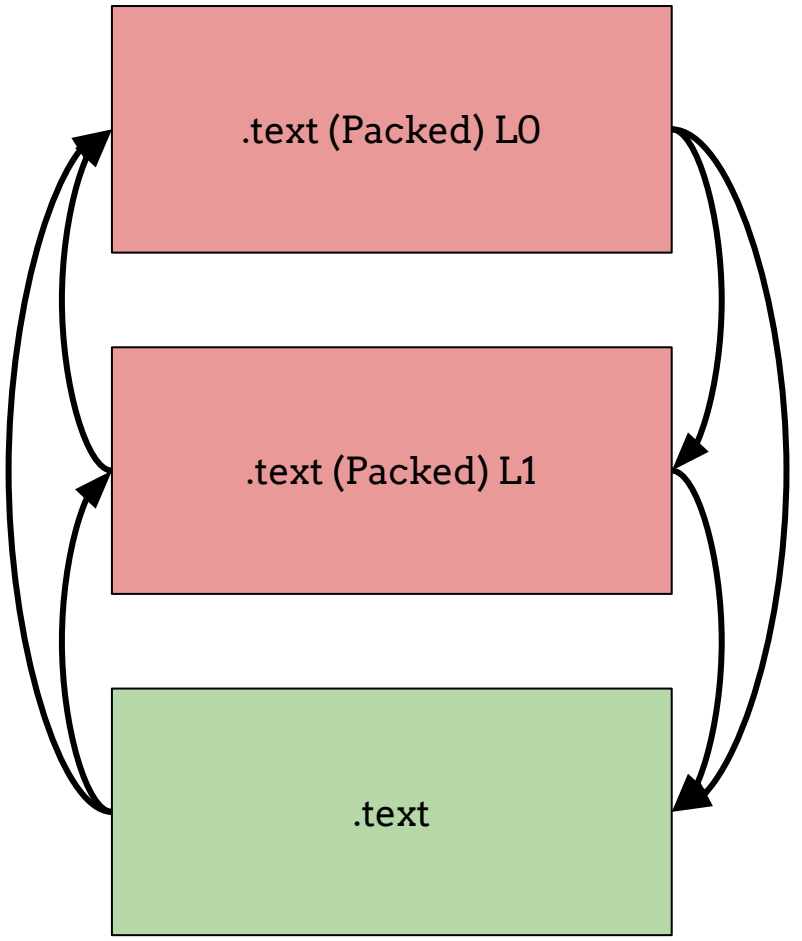
Linear: Type II



Cyclic: Type III



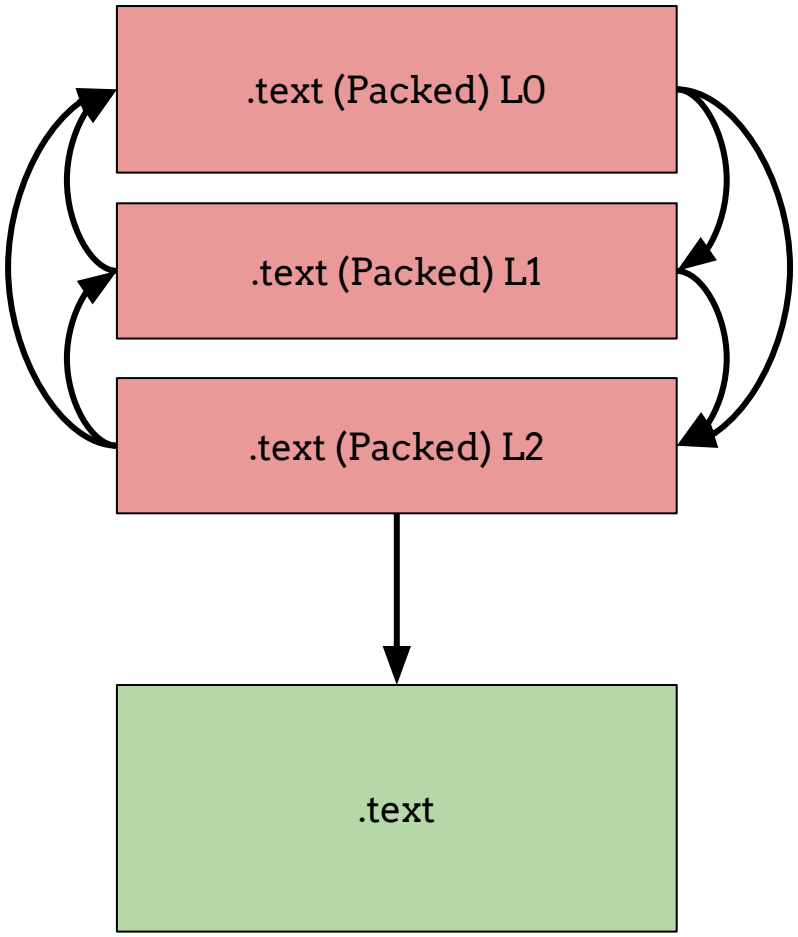
Interleaved: Type >= IV



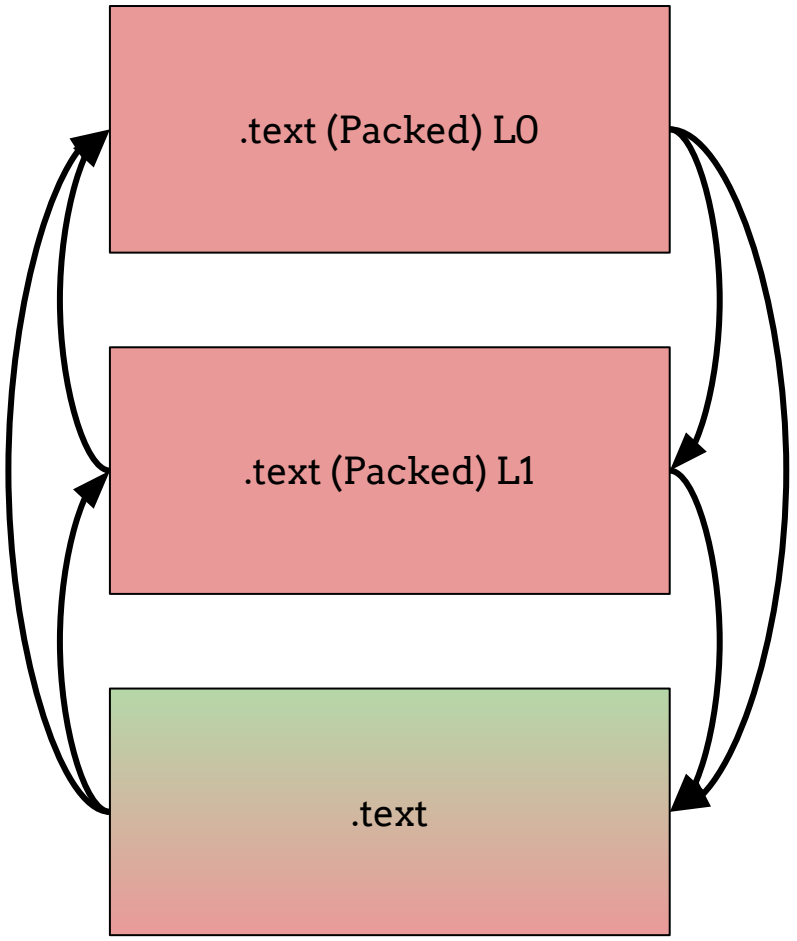
POLITECNICO
MILANO 1863

Code Visibility

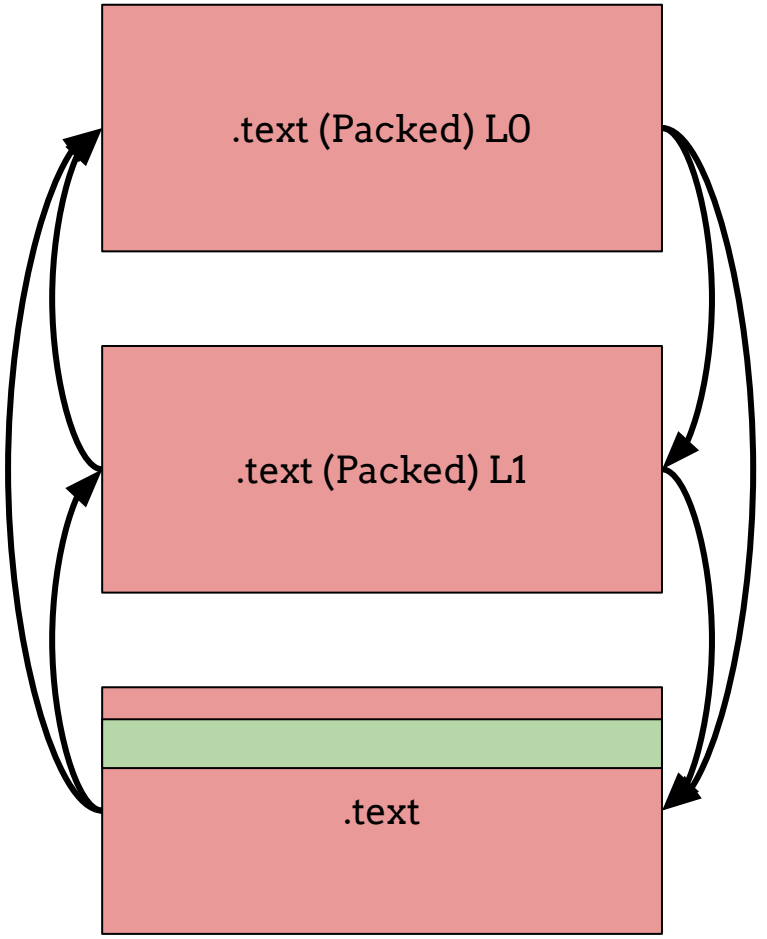
Full Code: Type IV



Incremental: Type V



Shift Frame: Type V



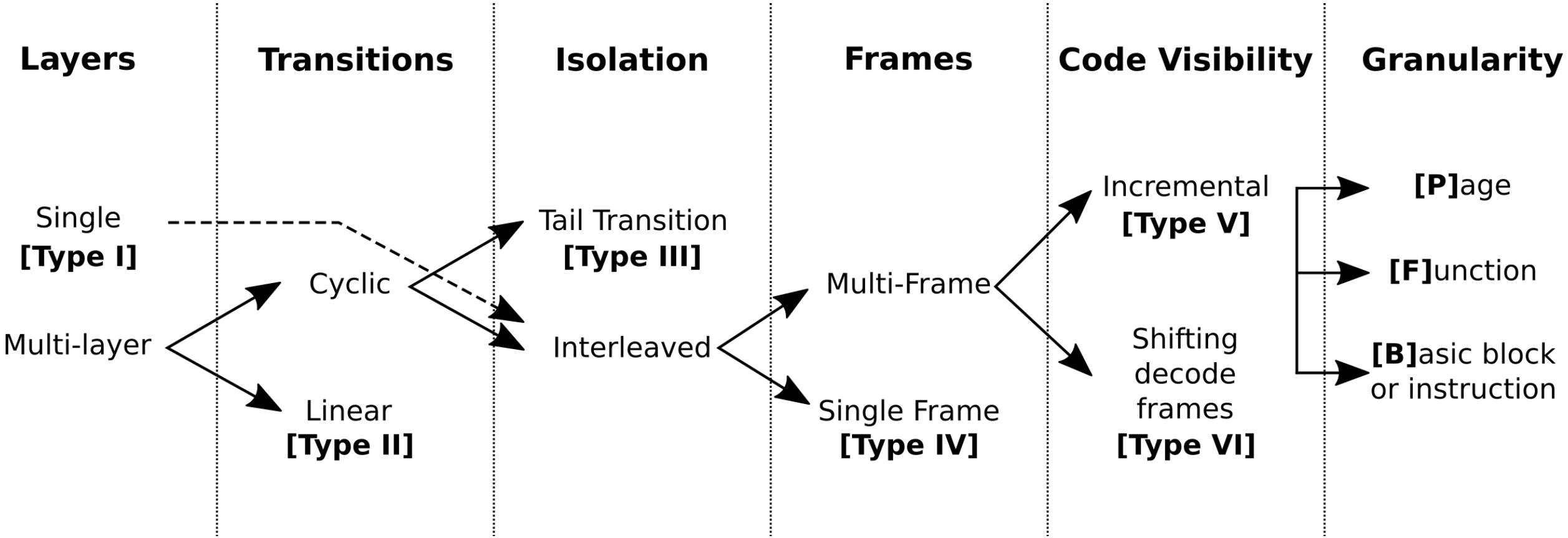
POLITECNICO
MILANO 1863

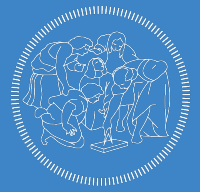
Granularity

- Pages
- Functions
- Basic Blocks
- Instructions

POLITECNICO
MILANO 1863

Packers Complexity





POLITECNICO
MILANO 1863

Examples of Packers

Type		Packer
I	Single Layer	UPX
II	Linear Multi Layer	Custom Packer
III	Cyclic Multi Layer	Themida
IV	Interleaved Single Frame	Upack
V	Incremental Multi-Frame	Beria
VI	Shifting Frames	Armadillo

POLITECNICO
MILANO 1863

Usage of Packers

Type	Off-the-Shelf	Custom Packer
I	25.3 % (173)	7.3% (443)
II	8.2 % (56)	12.4% (752)
III	51.4 % (352)	65.6 % (3993)
IV	12.6% (86)	13.8% (843)
V	0.9% (6)	0.8% (46)
VI	1.8% (12)	0.2% (11)

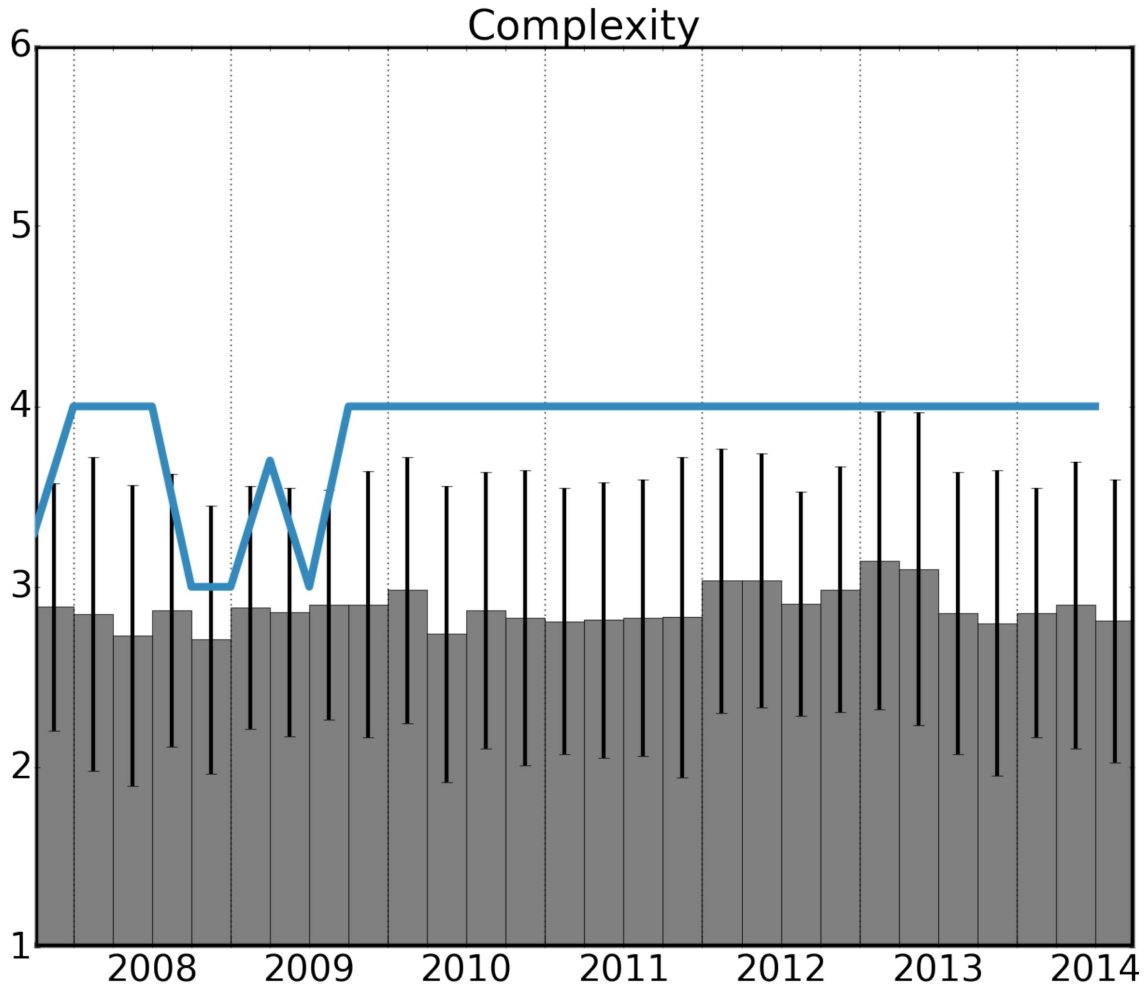
POLITECNICO
MILANO 1863

Usage of Packers

Type	Off-the-Shelf	Custom Packer
I	25.3 % (173)	7.3% (443)
II	8.2 % (56)	12.4% (752)
III	51.4 % (352)	65.6 % (3993)
IV	12.6% (86)	13.8% (843)
V	0.9% (6)	0.8% (46)
VI	1.8% (12)	0.2% (11)

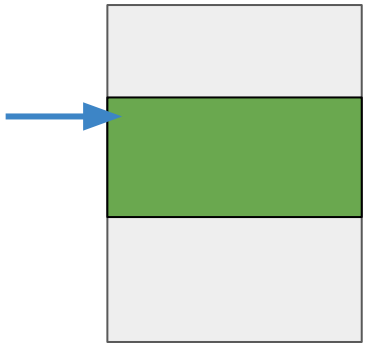
POLITECNICO
MILANO 1863

Packers Timeline

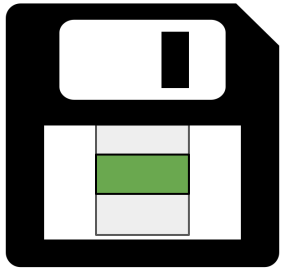


POLITECNICO
MILANO 1863

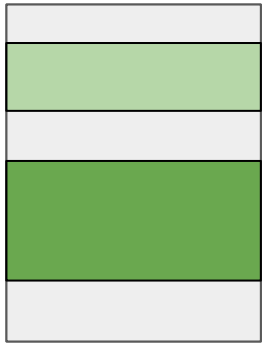
Unpacking Approach



Detect W and X memory regions

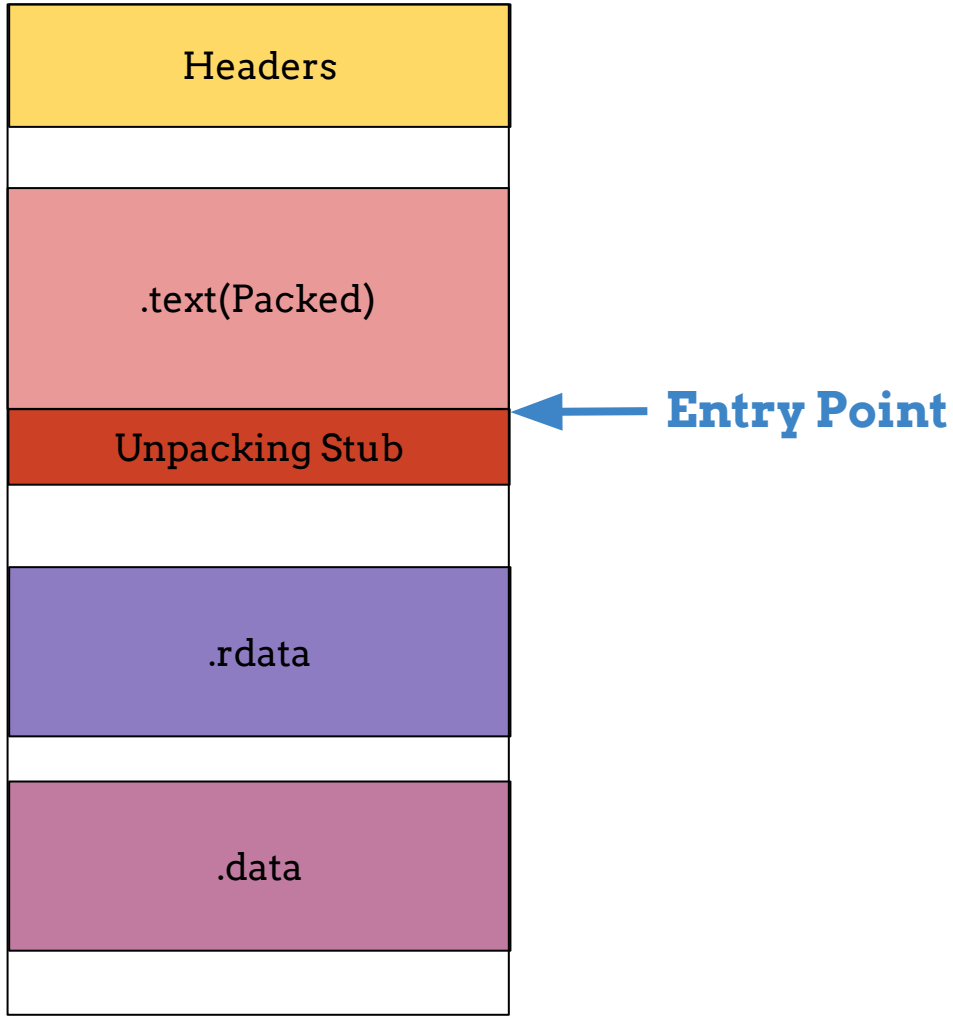


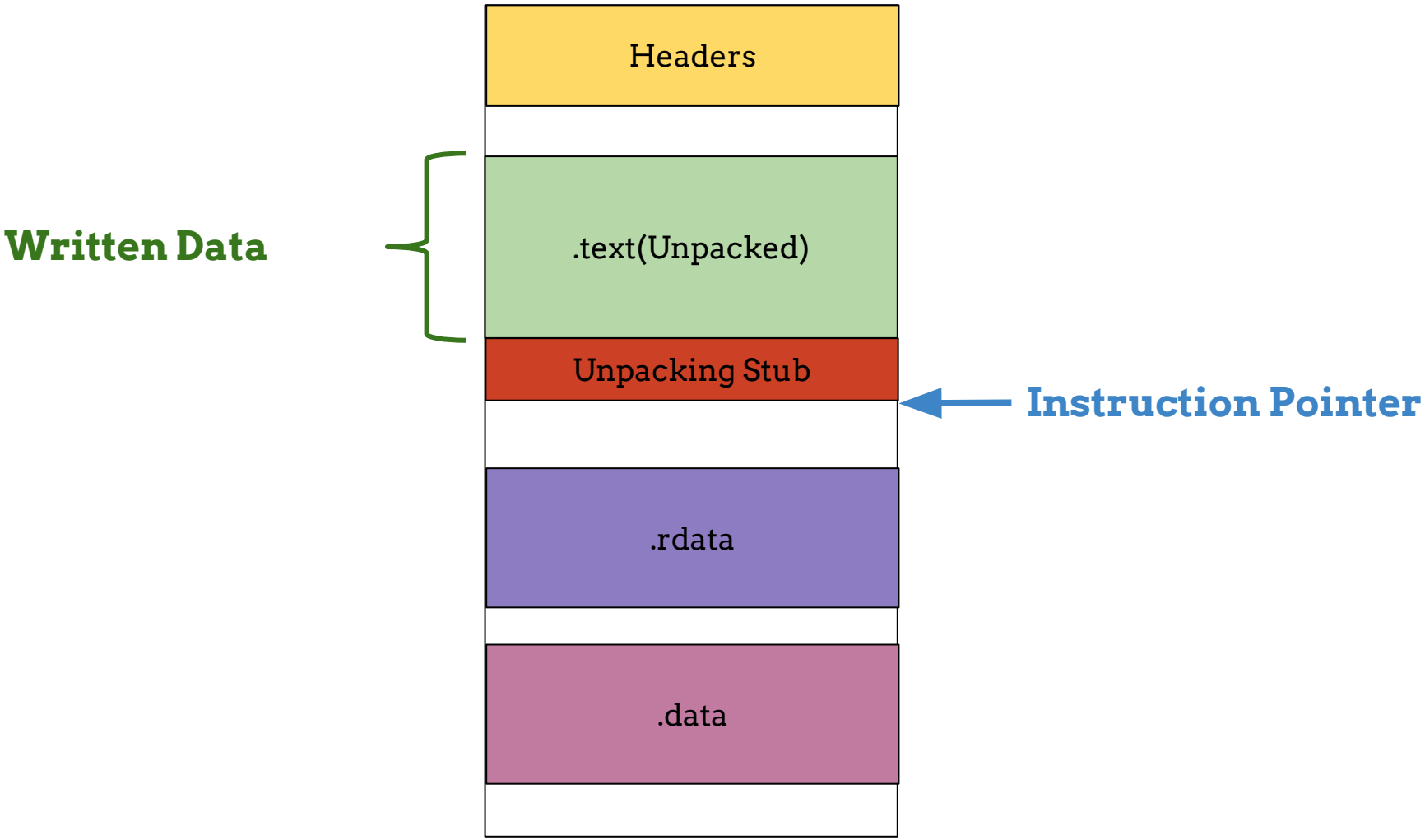
Dump the Program

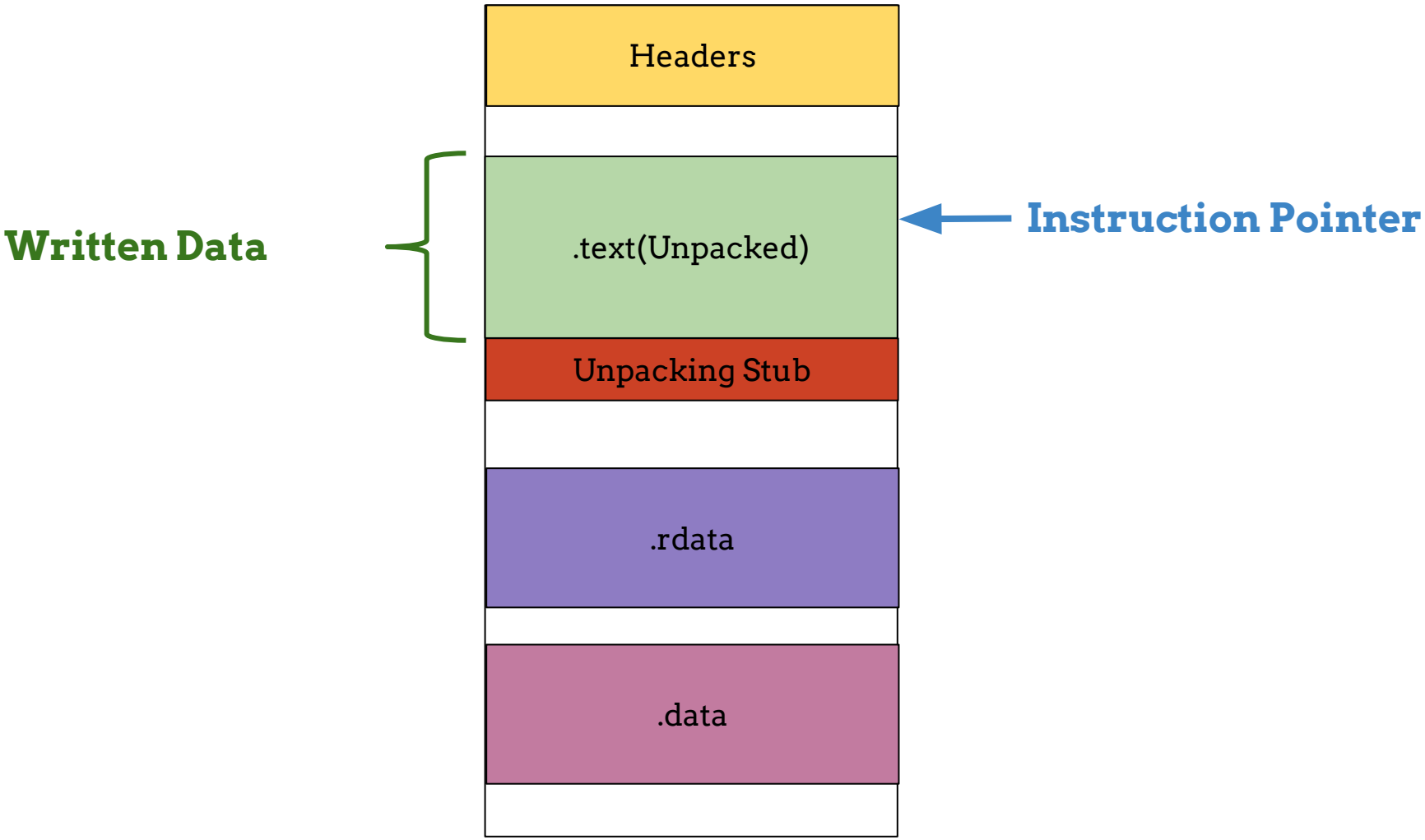


Deobfuscate the Import Address Table

Recognize the correct dump

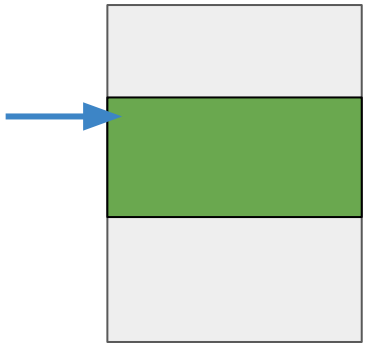

Headers
.text(Packed)
Unpacking Stub
Entry Point
.rdata
.data


Written Data
Headers
.text(Unpacked)
Unpacking Stub
Instruction Pointer
.rdata
.data

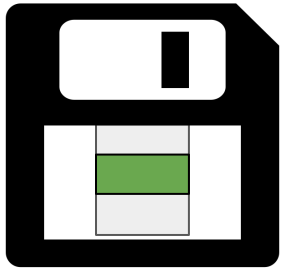

Written Data
Headers
.text(Unpacked)
Instruction Pointer
Unpacking Stub
.rdata
.data

POLITECNICO
MILANO 1863

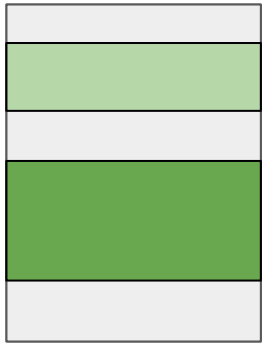
Unpacking Approach



Detect W and X memory regions

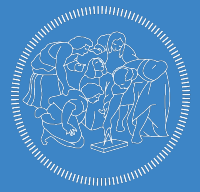


Dump the Program



Deobfuscate the Import Address Table

Recognize the correct dump



POLITECNICO MILANO 1863

Experiment 1 : known packers

	Upx	FSG	Mew	mpress	PeCompact	Obsidium	ExePacker	ezip
MessageBox.exe	✓	✓	✓	✓	✓	!	✓	✓
WinRAR.exe	✓	✓	✓	✓	✓	!	✓	✓

	Xcomp	PElock	ASProtect	ASPack	eXpressor	exe32packer	beropacker	Hyperion
MessageBox.exe	✓	!	!	✓	!	✓	✓	✓
WinRAR.exe	✓	!	!	✓	!	✓	✓	✓

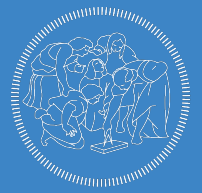
! → Original code dumped but Import directory not reconstructed

POLITECNICO MILANO 1863

Experiment 2 : wild samples

Number of packed (checked manually) samples 1096

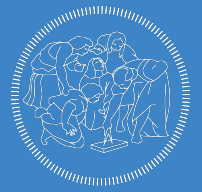
	N°	%
Unpacked and working	669	63
Unpacked but not executable	139	13
Not unpacked	258	24



POLITECNICO
MILANO 1863

Other Anti-Static Analysis

- **String Obfuscation**
- **IAT Obfuscation**
- **Dynamic Loading**
- **etc.**



POLITECNICO
MILANO 1863

Malware Evasive

```
If (amIUnderAnalysis())
{
    die();
}
else
{
   beMalicious();
}
```

Malware Evasive

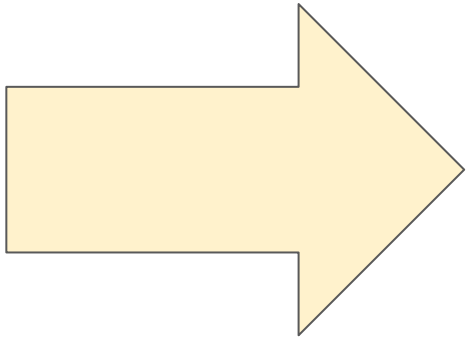
Artifacts

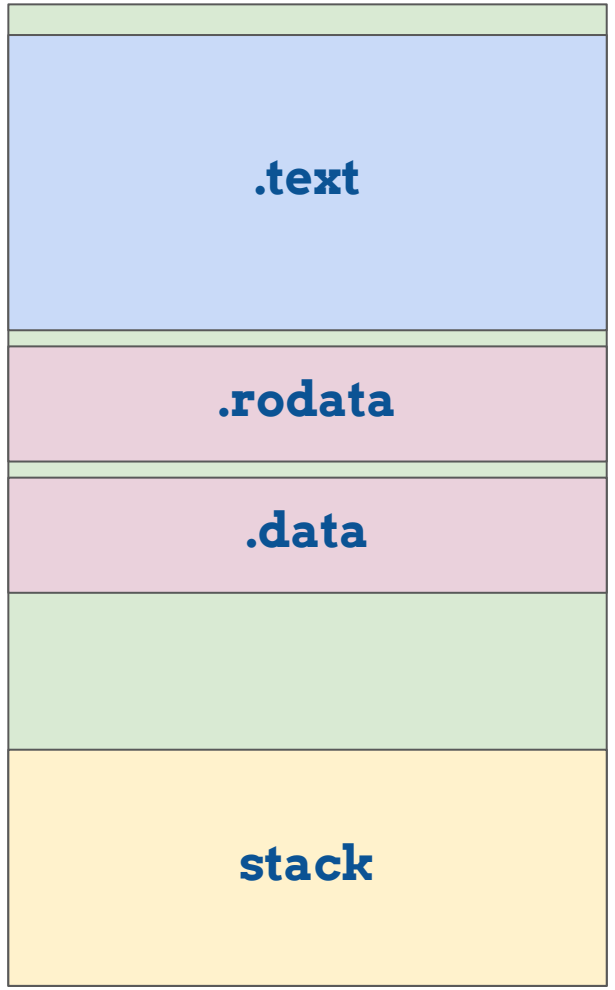
```
If (amIUnderAnalysis())
{
    die();
}
else
{
   beMalicious();
}
```

Dynamic Binary Instrumentation

POLITECNICO
MILANO 1863

What is a DBI Tool?

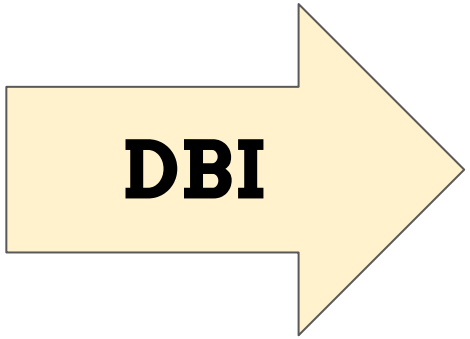


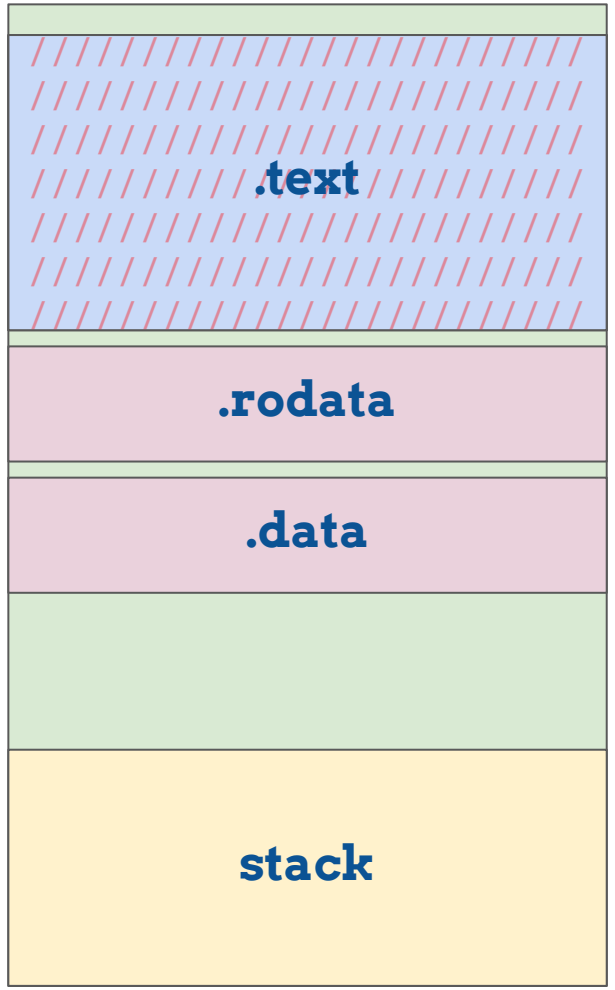


Memory

POLITECNICO
MILANO 1863

What is a DBI Tool?





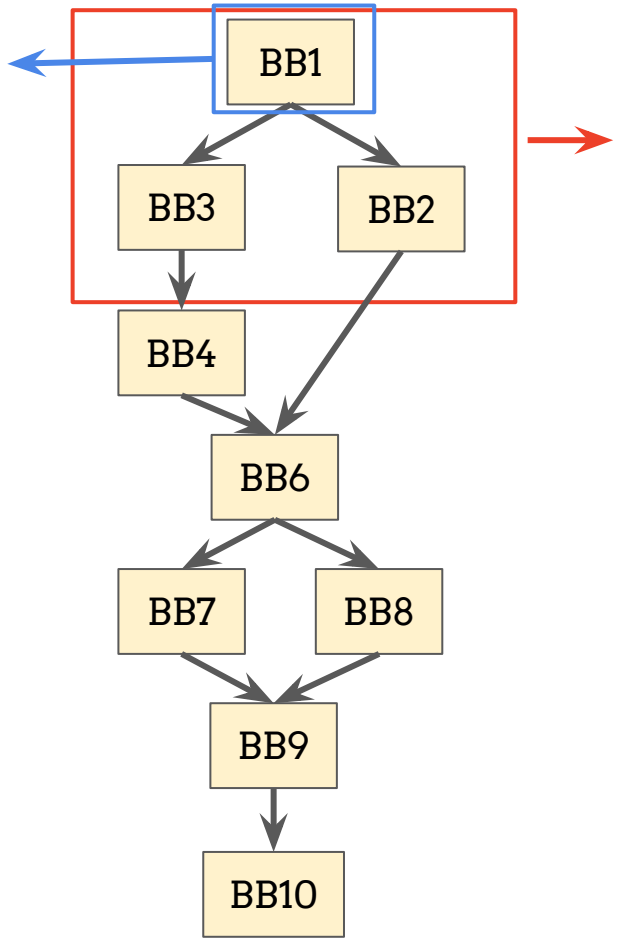
Memory

What is a DBI Tool?

Basic Block

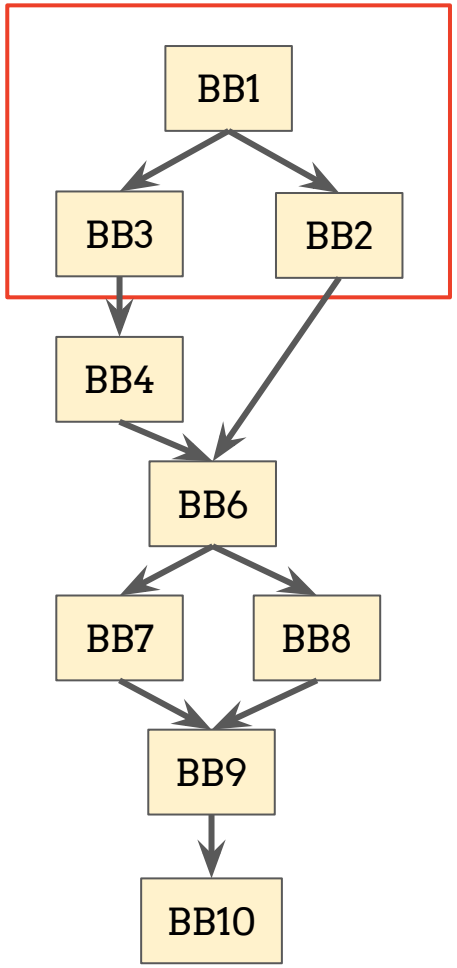
Trace

Control Flow Graph

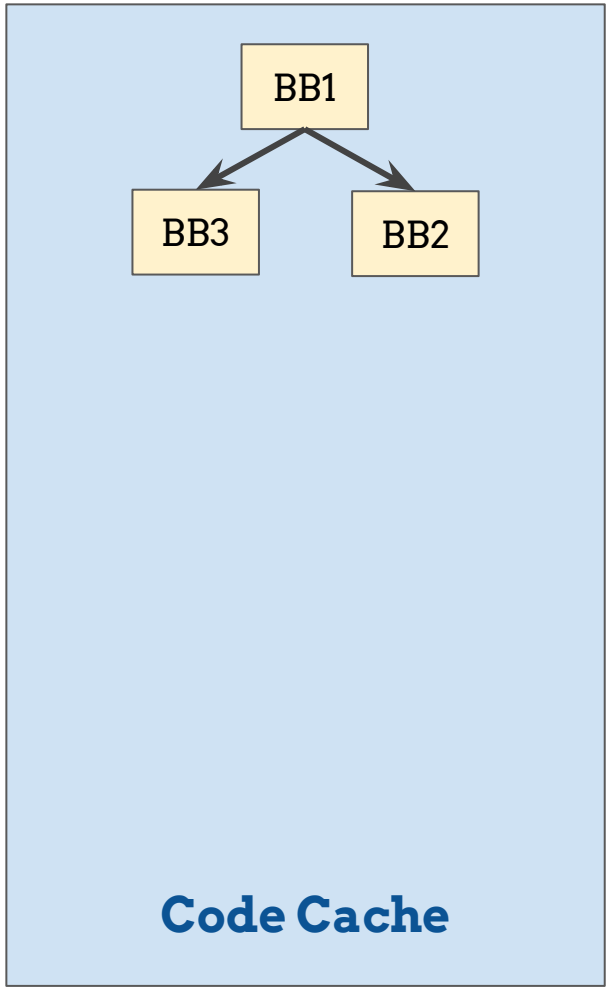


POLITECNICO
MILANO 1863

What is a DBI Tool?

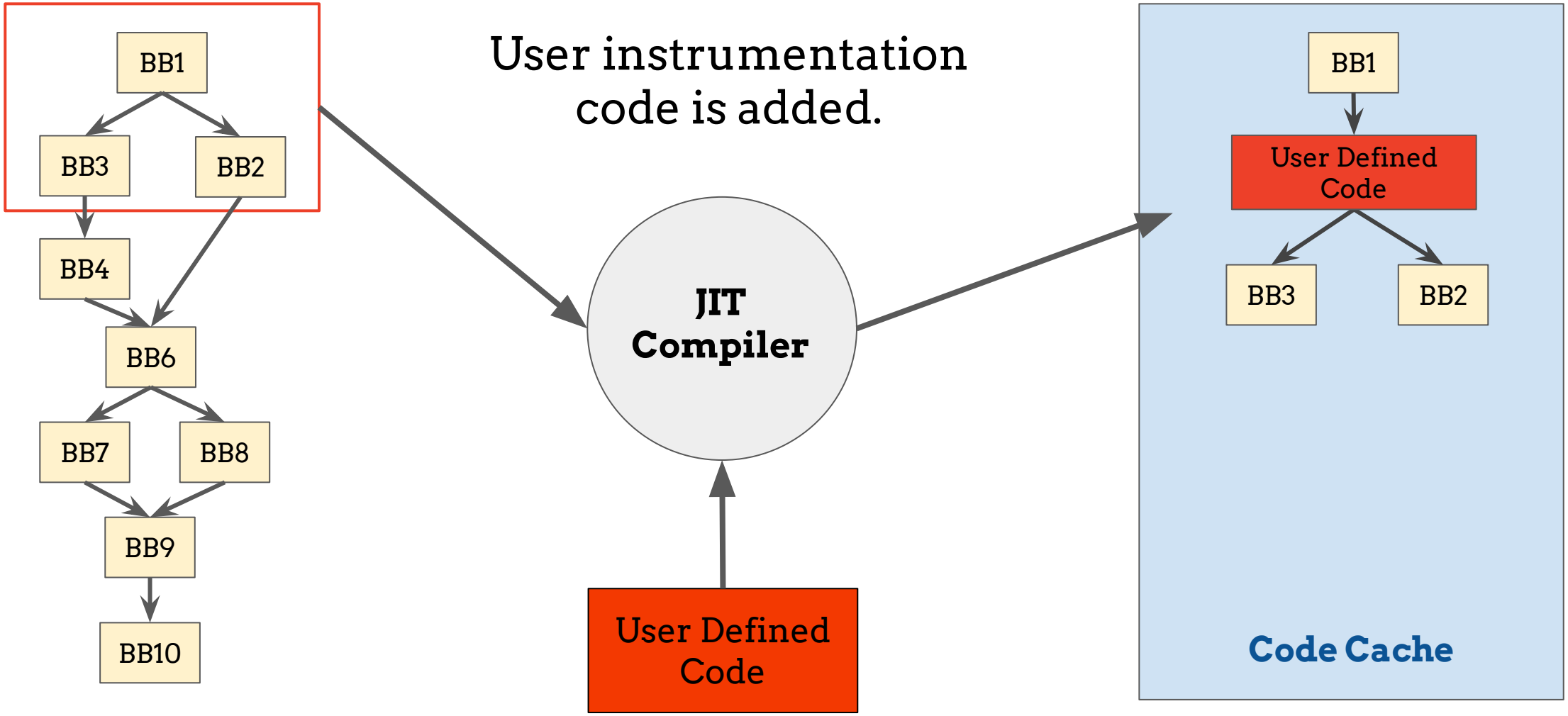


Trace is copied in the code cache



POLITECNICO
MILANO 1863

What is a DBI Tool?



POLITECNICO
MILANO 1863

DBI - Evasive Malware

Intel Pin Tools

DynamoRIO

Valgrind

rev.ng

rev.ng

POLITECNICO
MILANO 1863

DBI - Evasive Malware

Intel Pin Tools

DynamoRIO

Valgrind

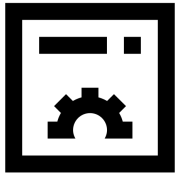
rev.ng

Artifacts

POLITECNICO
MILANO 1863

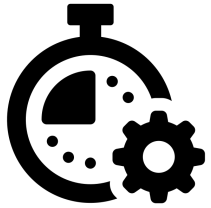
DBI - Evasive Malware

Code Cache Artifacts



JIT Compiler Detection

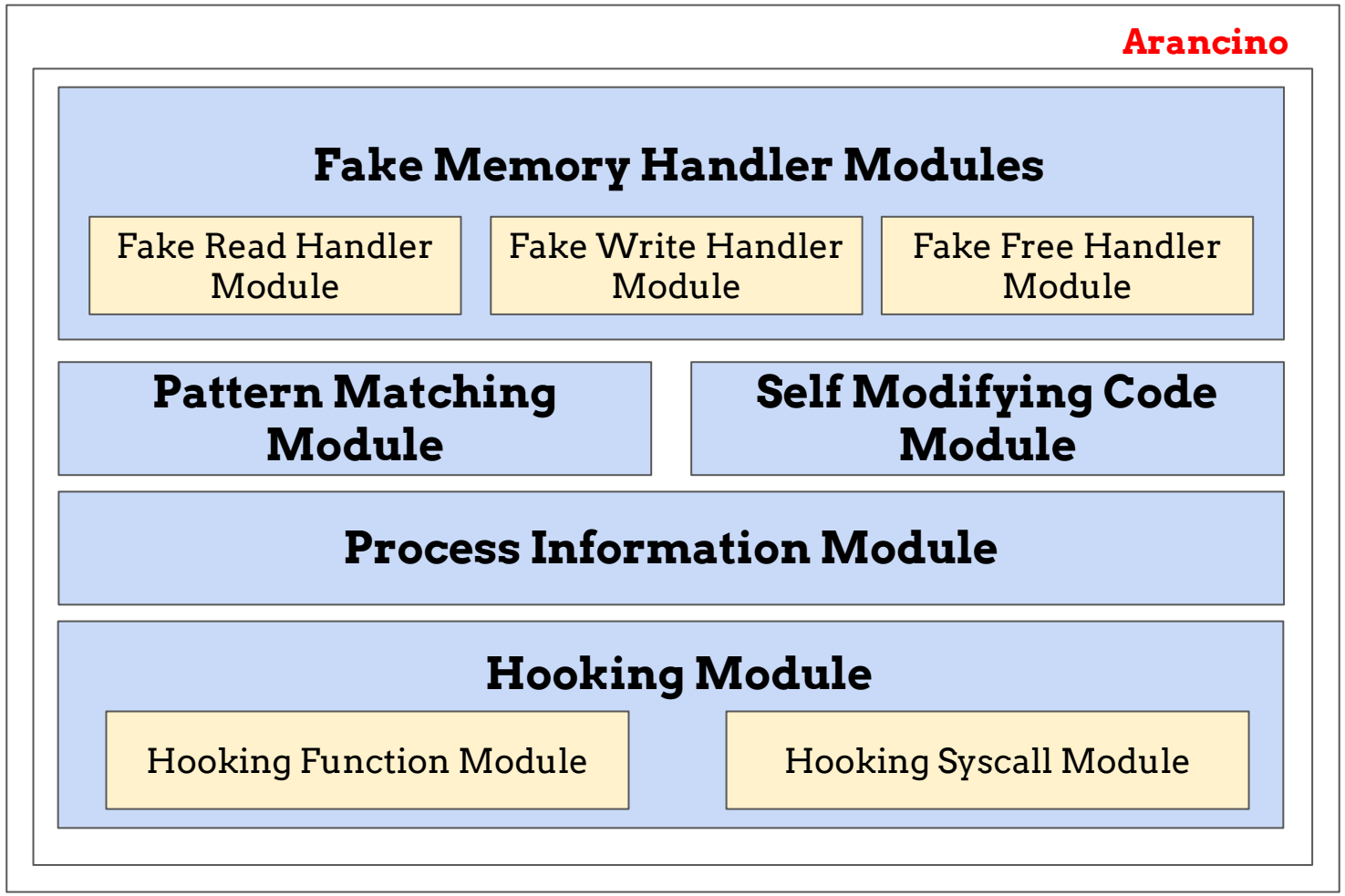
Environment Artifact



Overhead Detection

POLITECNICO
MILANO 1863

Arancino

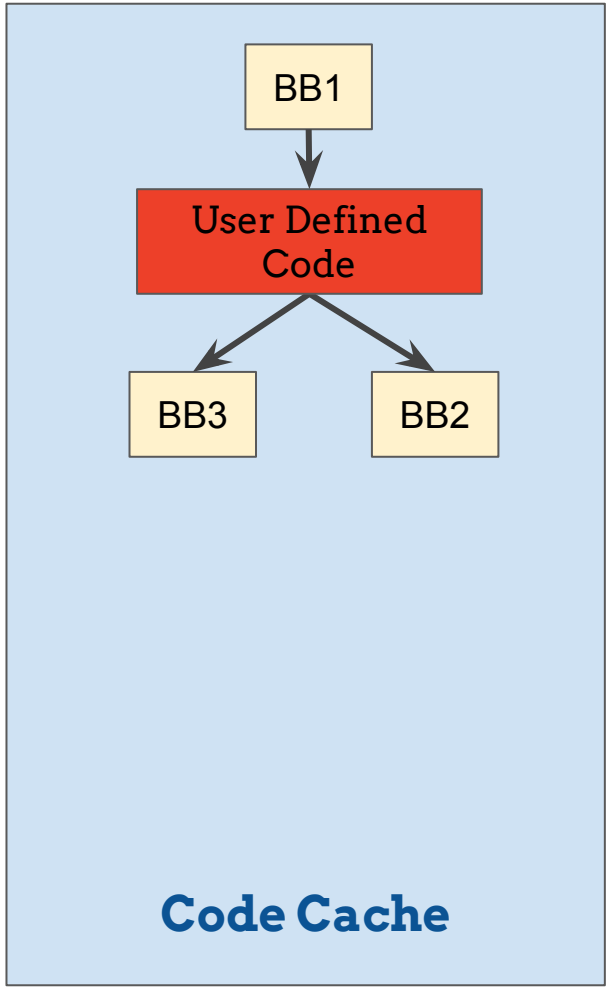


Code Cache Artifacts

Code Cache Artifacts

All those artifacts caused by having a Code Cache

- **IP Detection**
- **Self-Modifying Code**



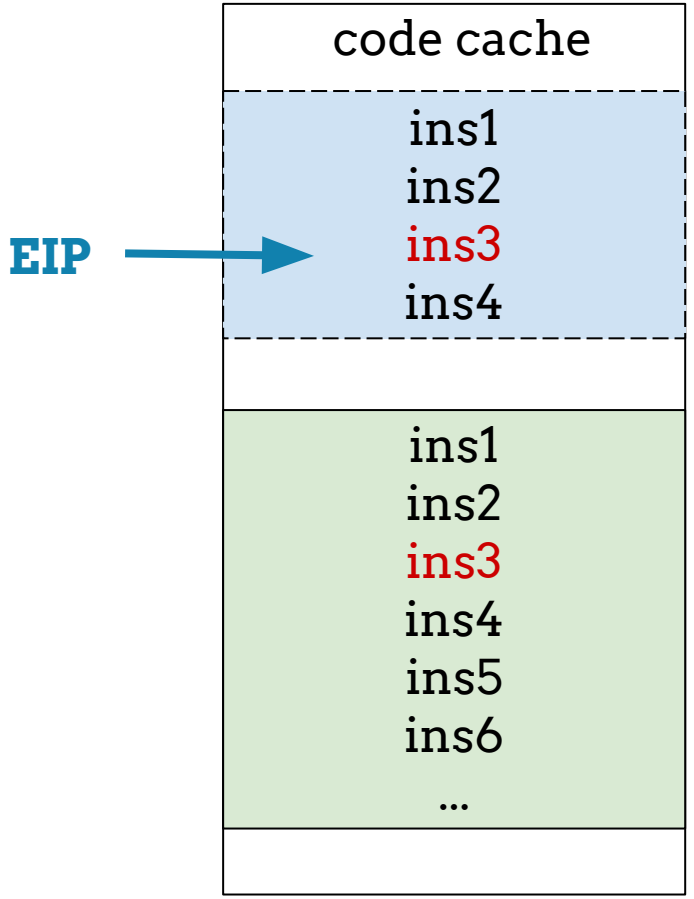
CCA - IP Detection

Nt Sycall (EIP -> EDX)

```
int 2e
```

Floating Point Context on the Stack

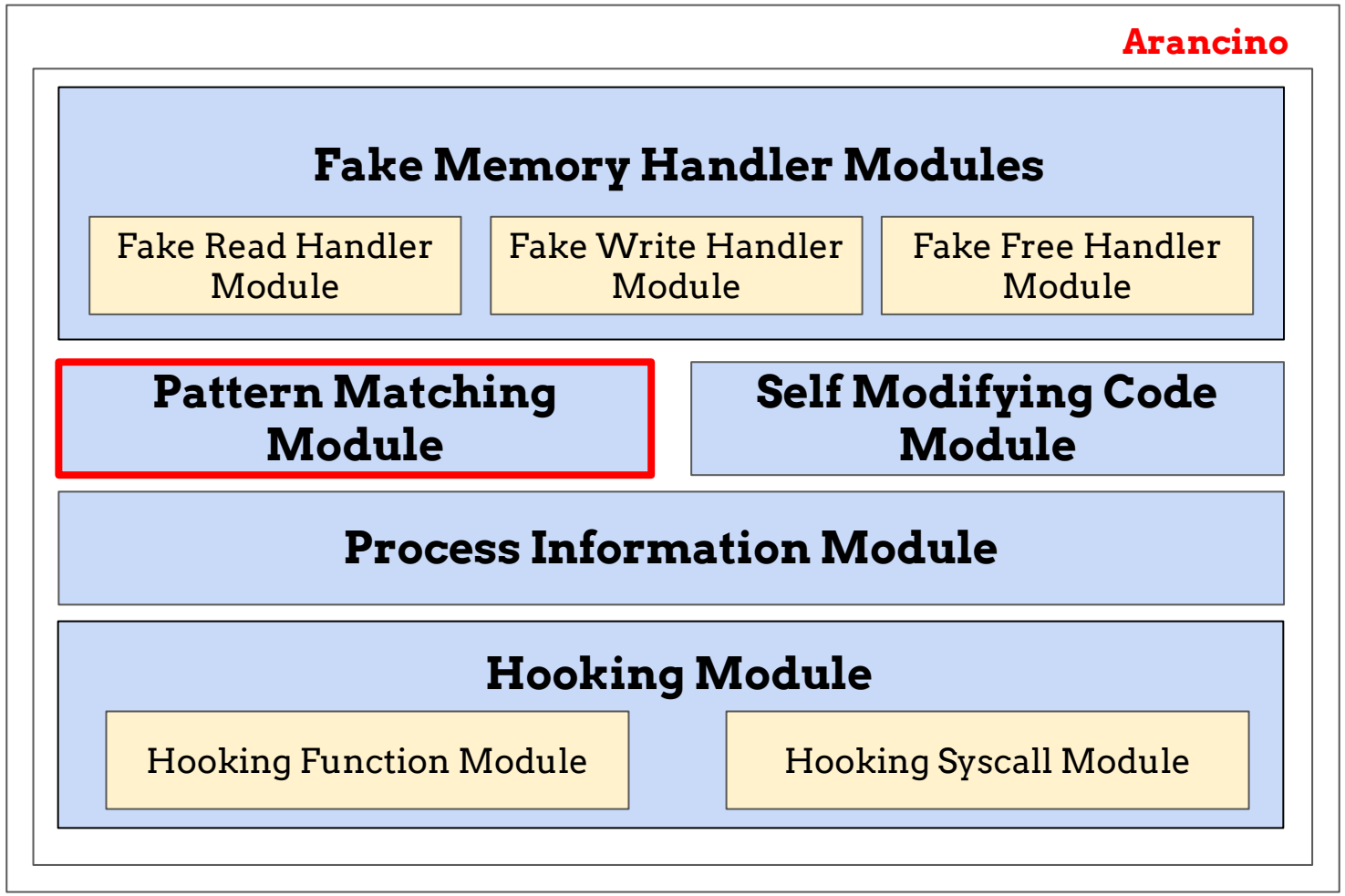
```
fsave/ fxsave/ fstenv
```



Arancino - Pattern Matching Module

- **PatchMap**: List of instructions and func pointers

- **PatchDispatcher**: check and add patch to instructions during trace building.



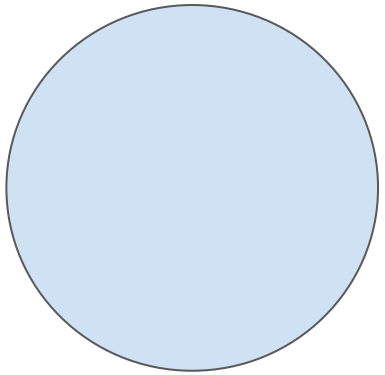
POLITECNICO
MILANO 1863

CCA - IP Detection

TRACE

add eax,4
int 2e
jmp 0x0804856c

PATCH DISPATCHER



int 2e
fsave
fxsave

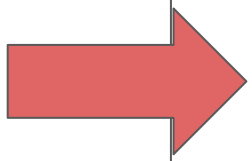
PATCHED TRACE

POLITECNICO
MILANO 1863

CCA - IP Detection

TRACE

add eax,4
int 2e
jmp 0x0804856c



PATCH DISPATCHER

add eax,4

int 2e
fsave
fxsave

PATCHED TRACE

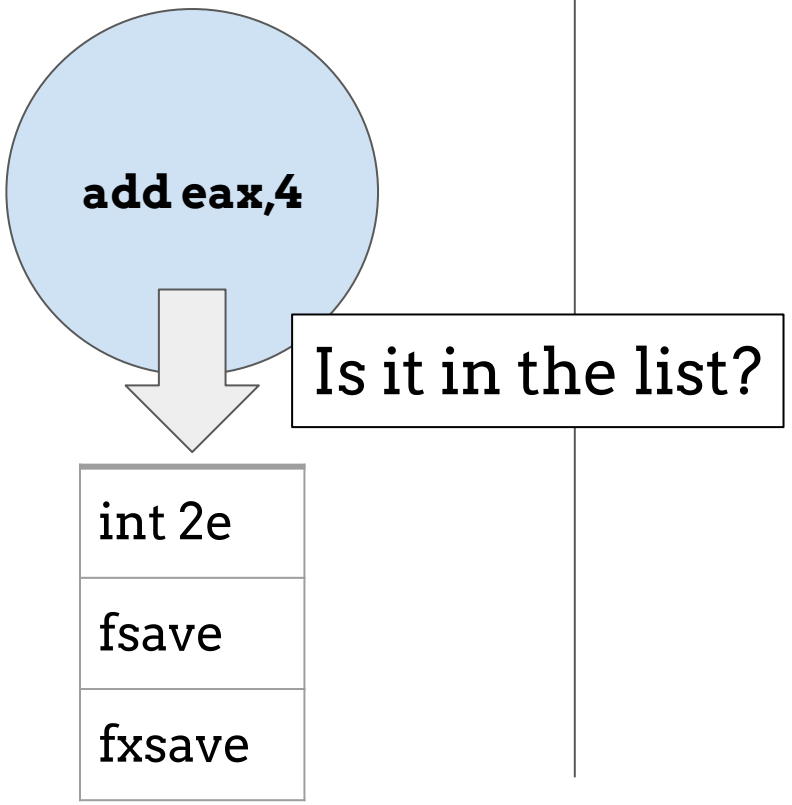
POLITECNICO
MILANO 1863

CCA - IP Detection

TRACE

add eax,4
int 2e
jmp 0x0804856c

PATCH DISPATCHER



PATCHED TRACE

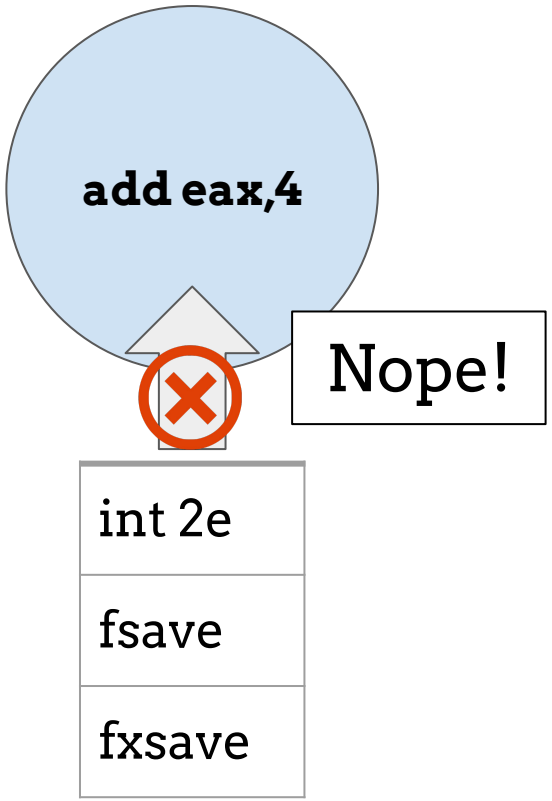
POLITECNICO
MILANO 1863

CCA - IP Detection

TRACE

add eax,4
int 2e
jmp 0x0804856c

PATCH DISPATCHER



PATCHED TRACE

POLITECNICO
MILANO 1863

CCA - IP Detection

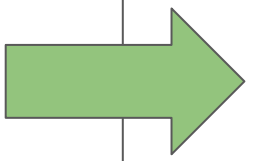
TRACE

add eax,4
int 2e
jmp 0x0804856c

PATCH DISPATCHER

add eax,4

int 2e
fsave
fxsave



PATCHED TRACE

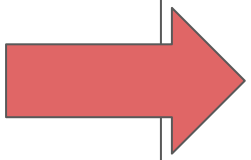
add eax,4

POLITECNICO
MILANO 1863

CCA - IP Detection

TRACE

add eax,4
int 2e
jmp 0x0804856c



PATCH DISPATCHER

int 2e

int 2e
fsave
fxsave

PATCHED TRACE

add eax,4

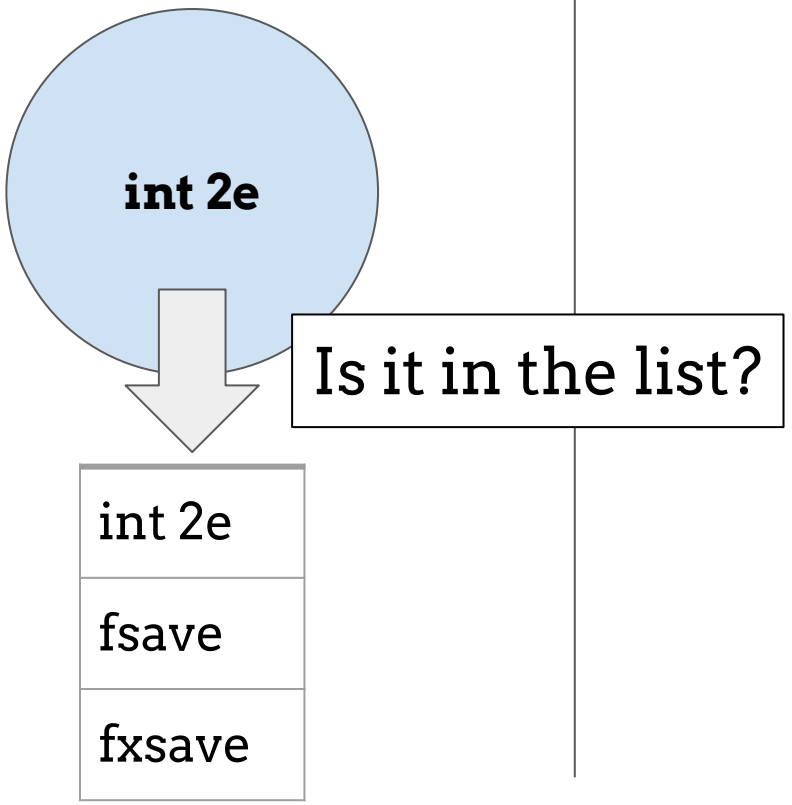
POLITECNICO
MILANO 1863

CCA - IP Detection

TRACE

add eax,4
int 2e
jmp 0x0804856c

PATCH DISPATCHER



PATCHED TRACE

add eax,4

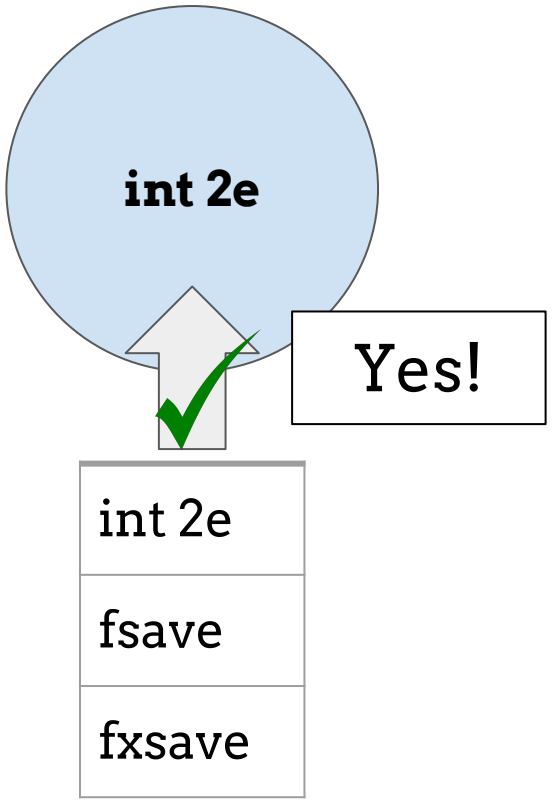
POLITECNICO
MILANO 1863

CCA - IP Detection

TRACE

add eax,4
int 2e
jmp 0x0804856c

PATCH DISPATCHER



PATCHED TRACE

add eax,4

POLITECNICO
MILANO 1863

CCA - IP Detection

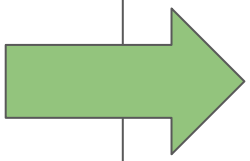
TRACE

add eax,4
int 2e
jmp 0x0804856c

PATCH DISPATCHER

int 2e

int 2e
fsave
fxsave

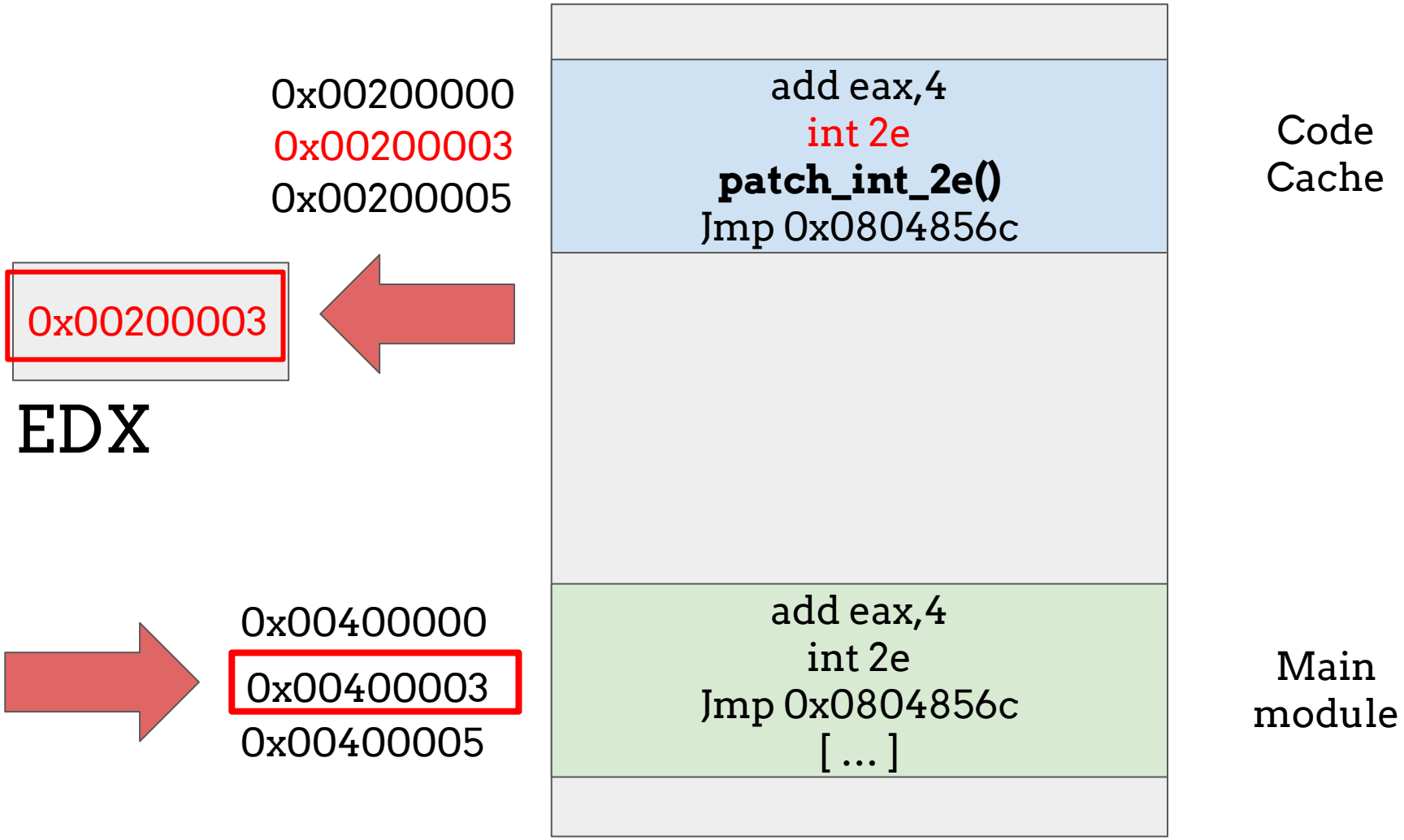


PATCHED TRACE

add eax,4
int 2e
patch_int2e()

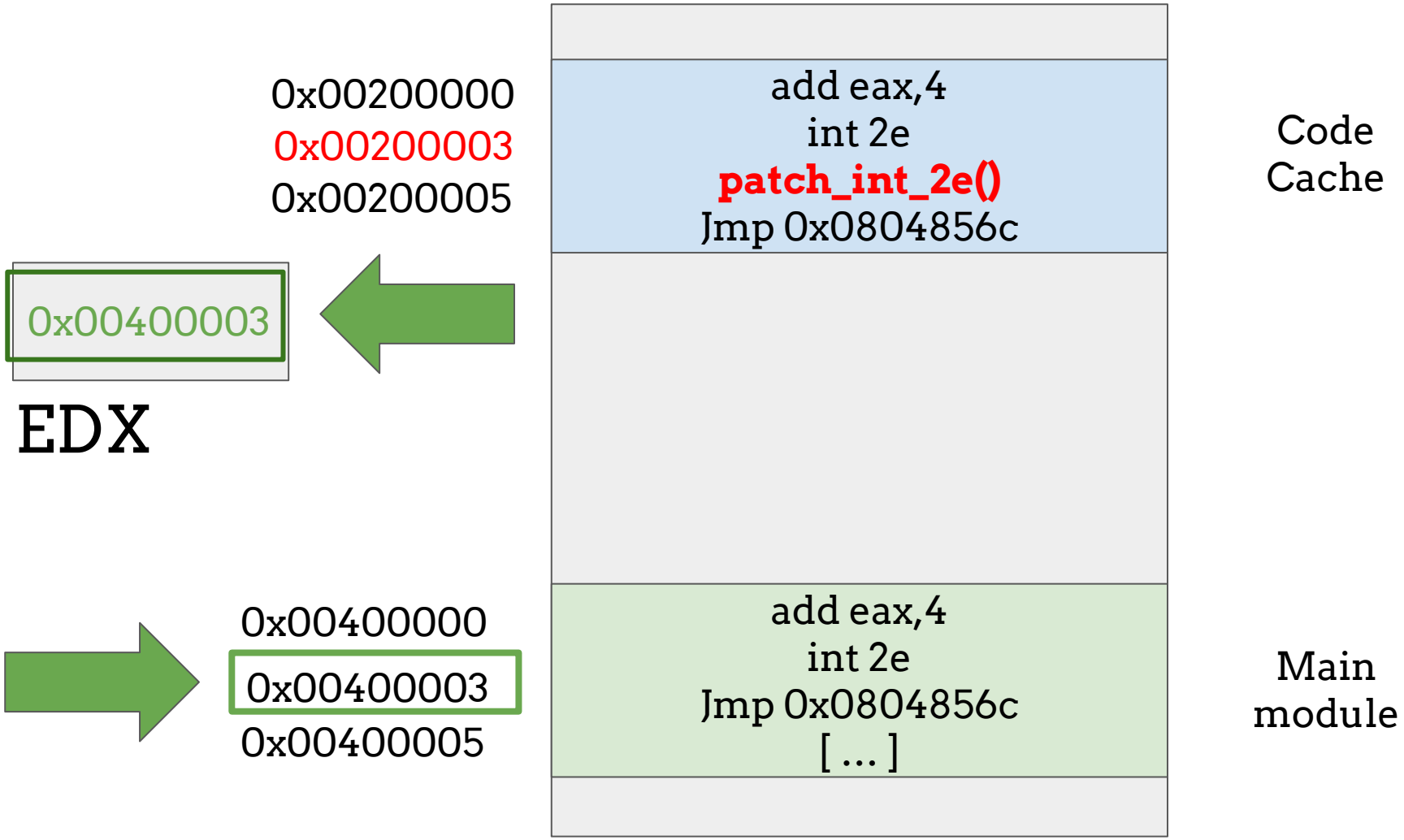
POLITECNICO
MILANO 1863

CCA - IP Detection RT



POLITECNICO
MILANO 1863

CCA - IP Detection RT

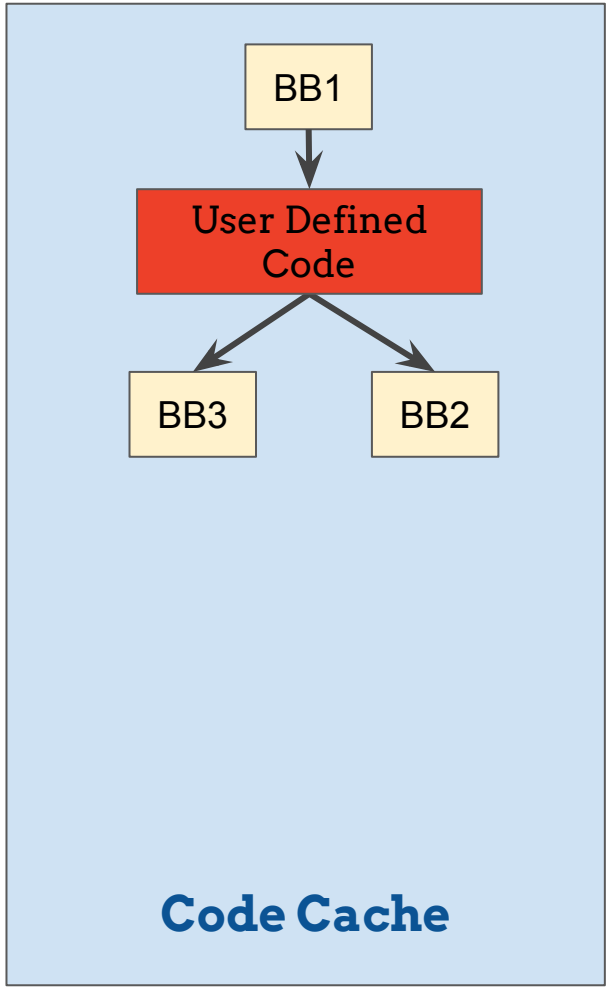


Code Cache Artifacts

All those artifacts caused by having a Code Cache

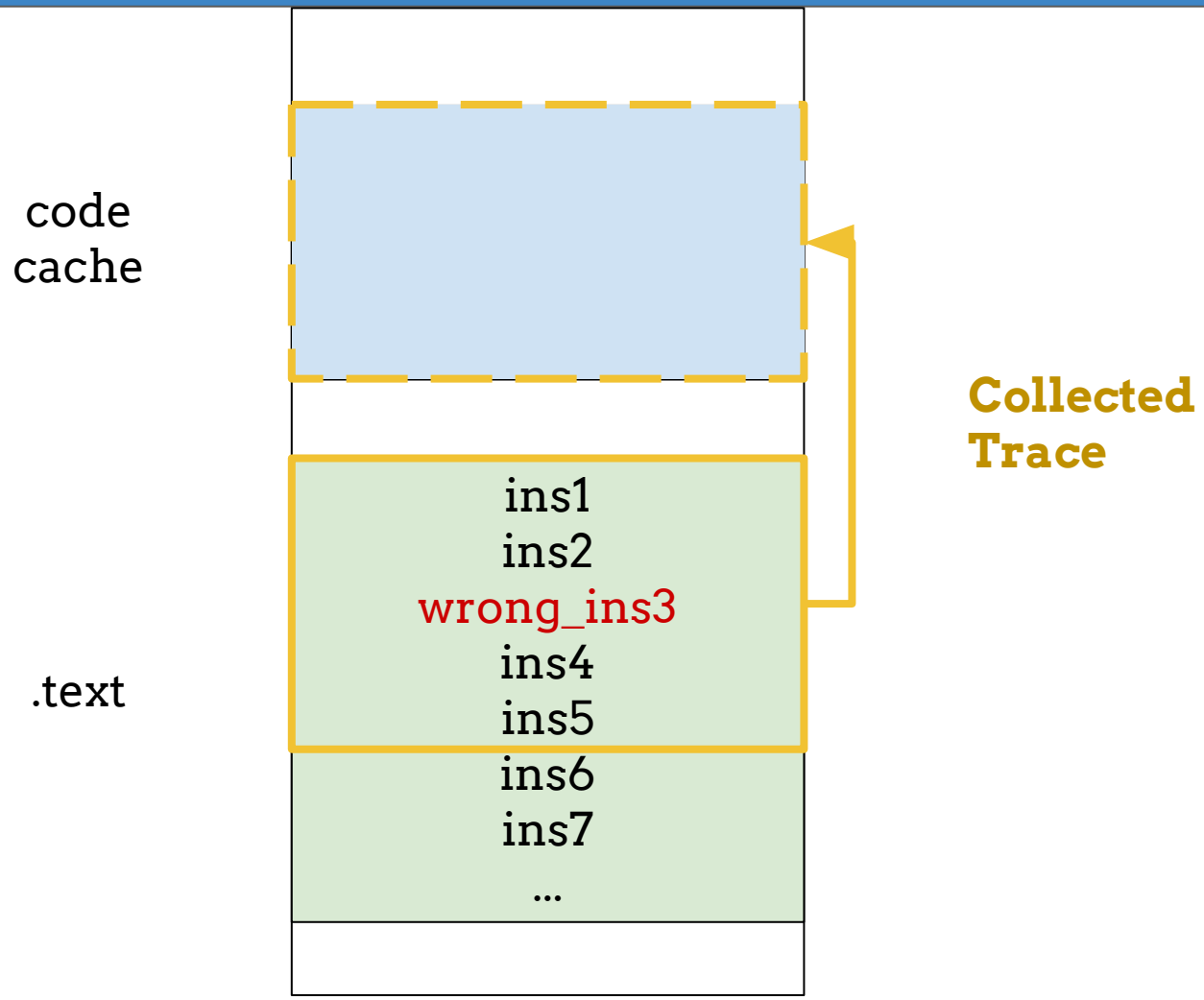
- **IP Detection**

- **Self-Modifying Code**



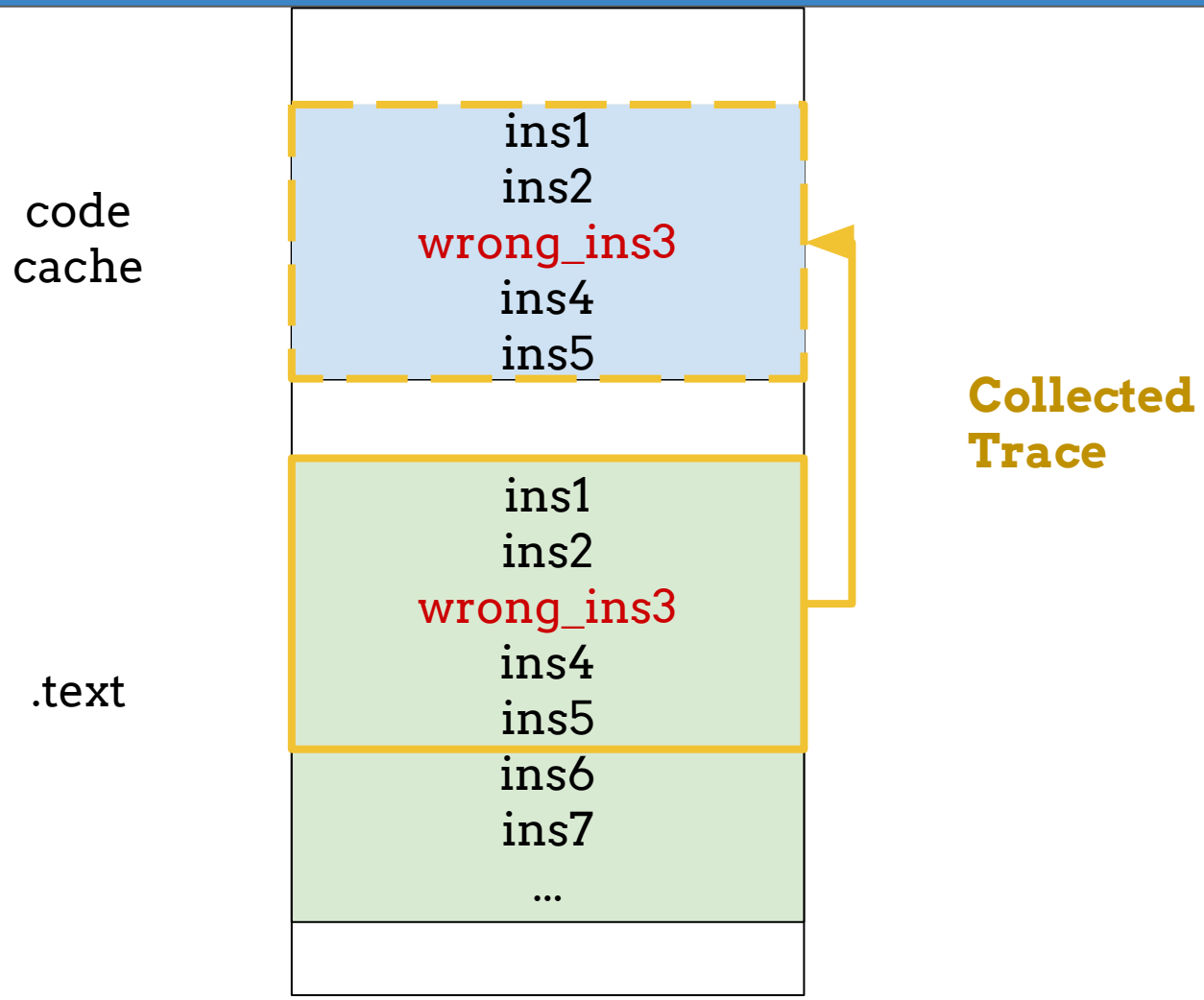
POLITECNICO
MILANO 1863

CCA - Self Modifying Code



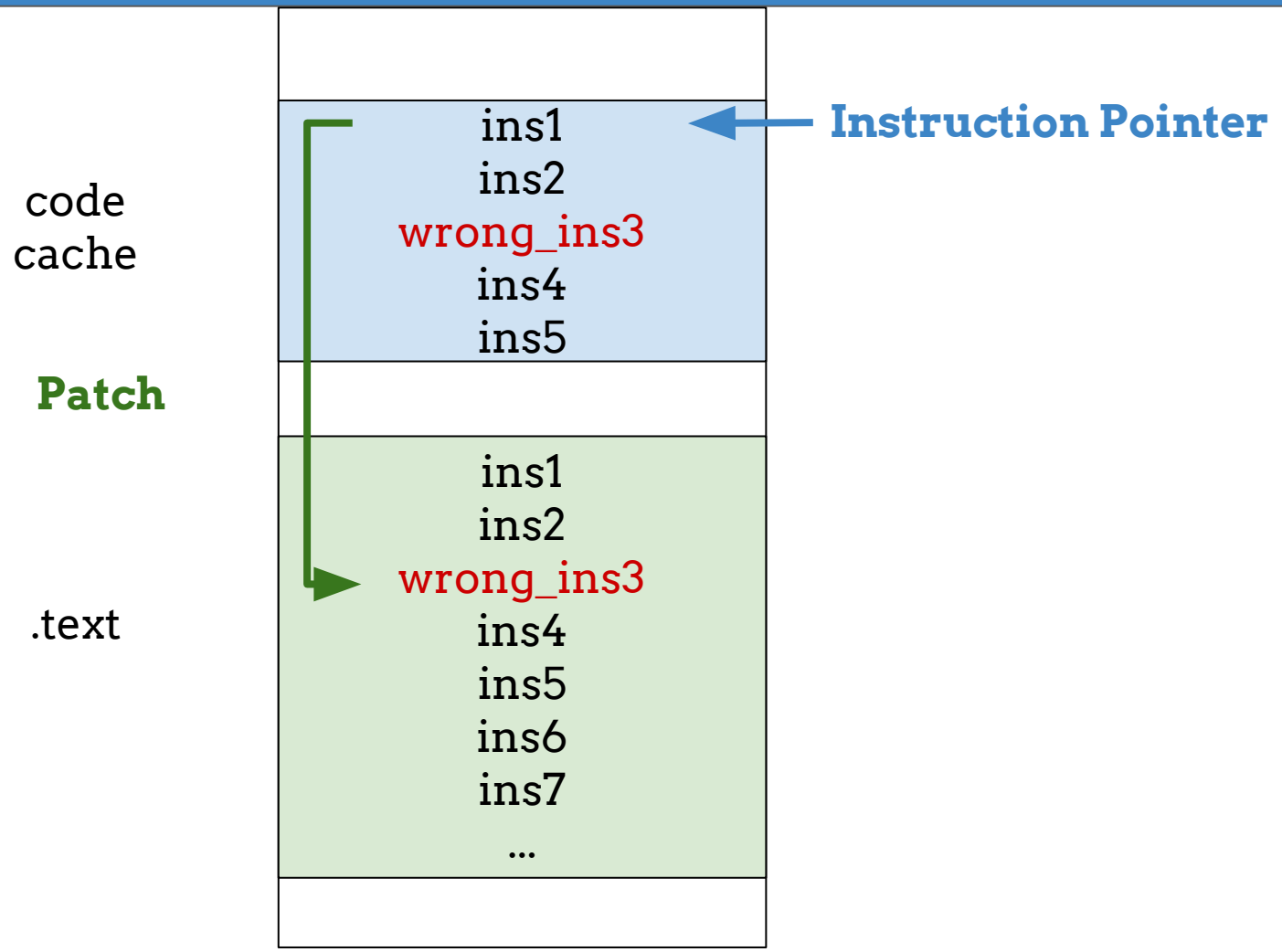
POLITECNICO MILANO 1863

CCA - Self Modifying Code



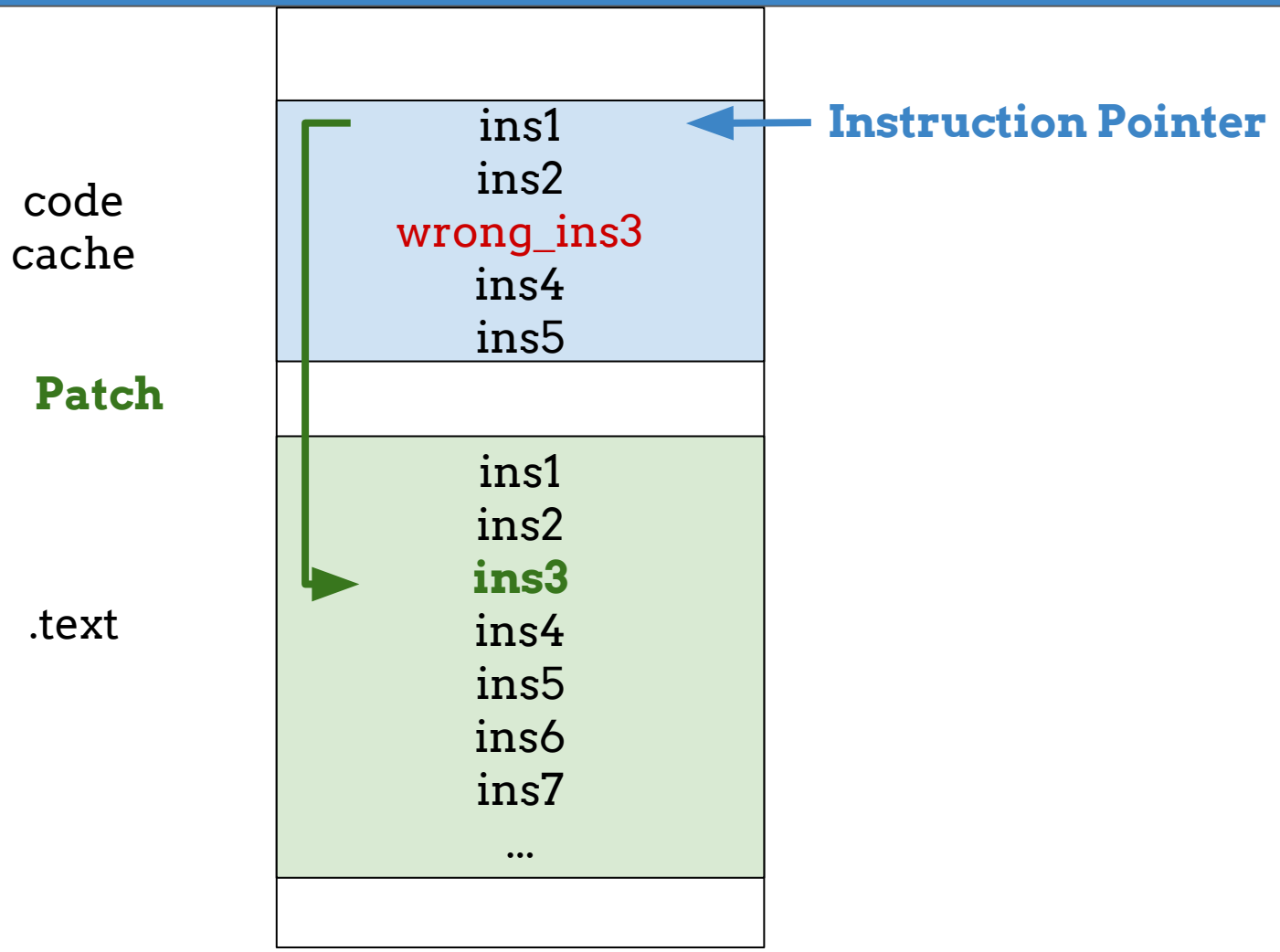
POLITECNICO MILANO 1863

CCA - Self Modifying Code

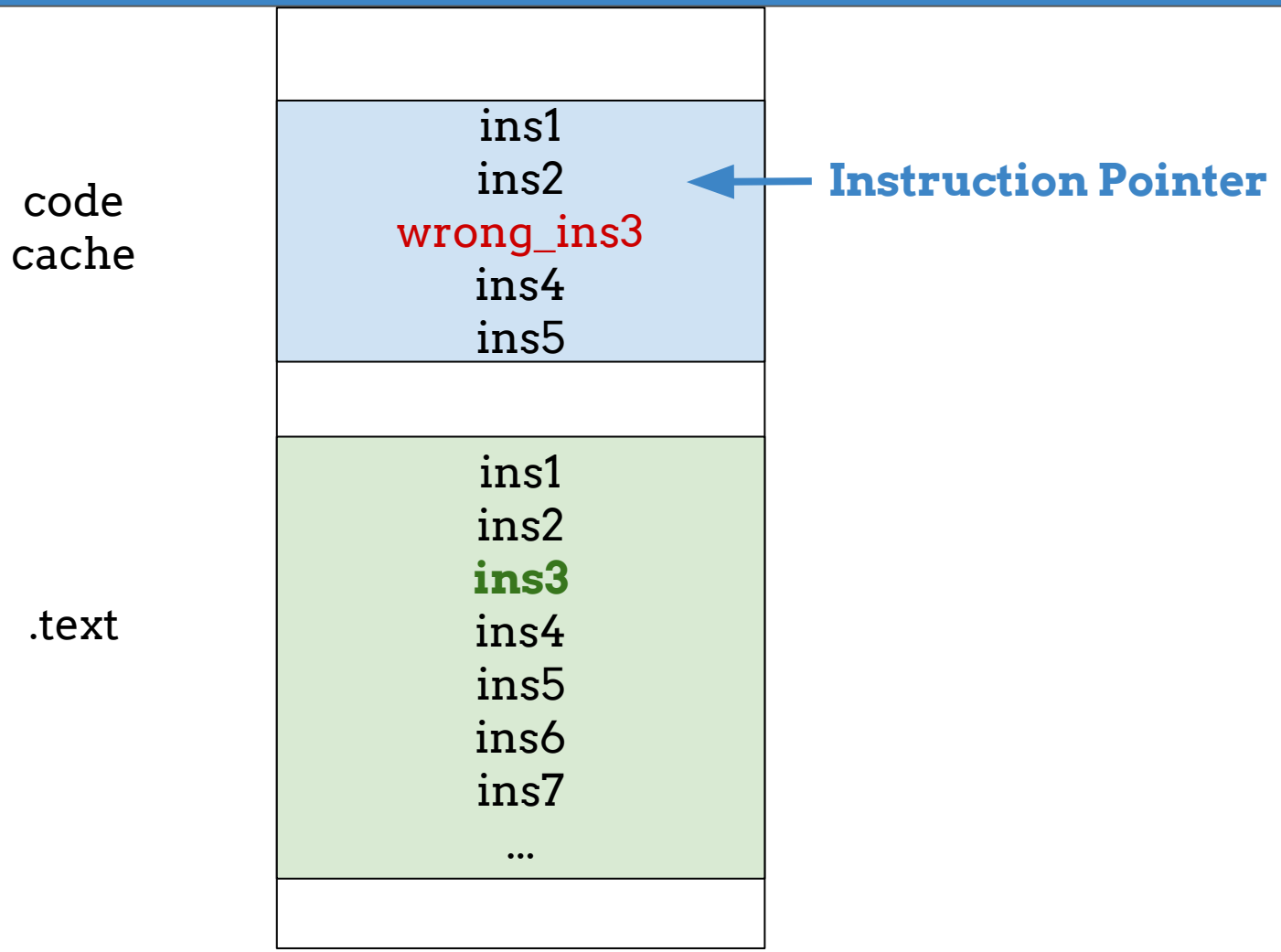


POLITECNICO MILANO 1863

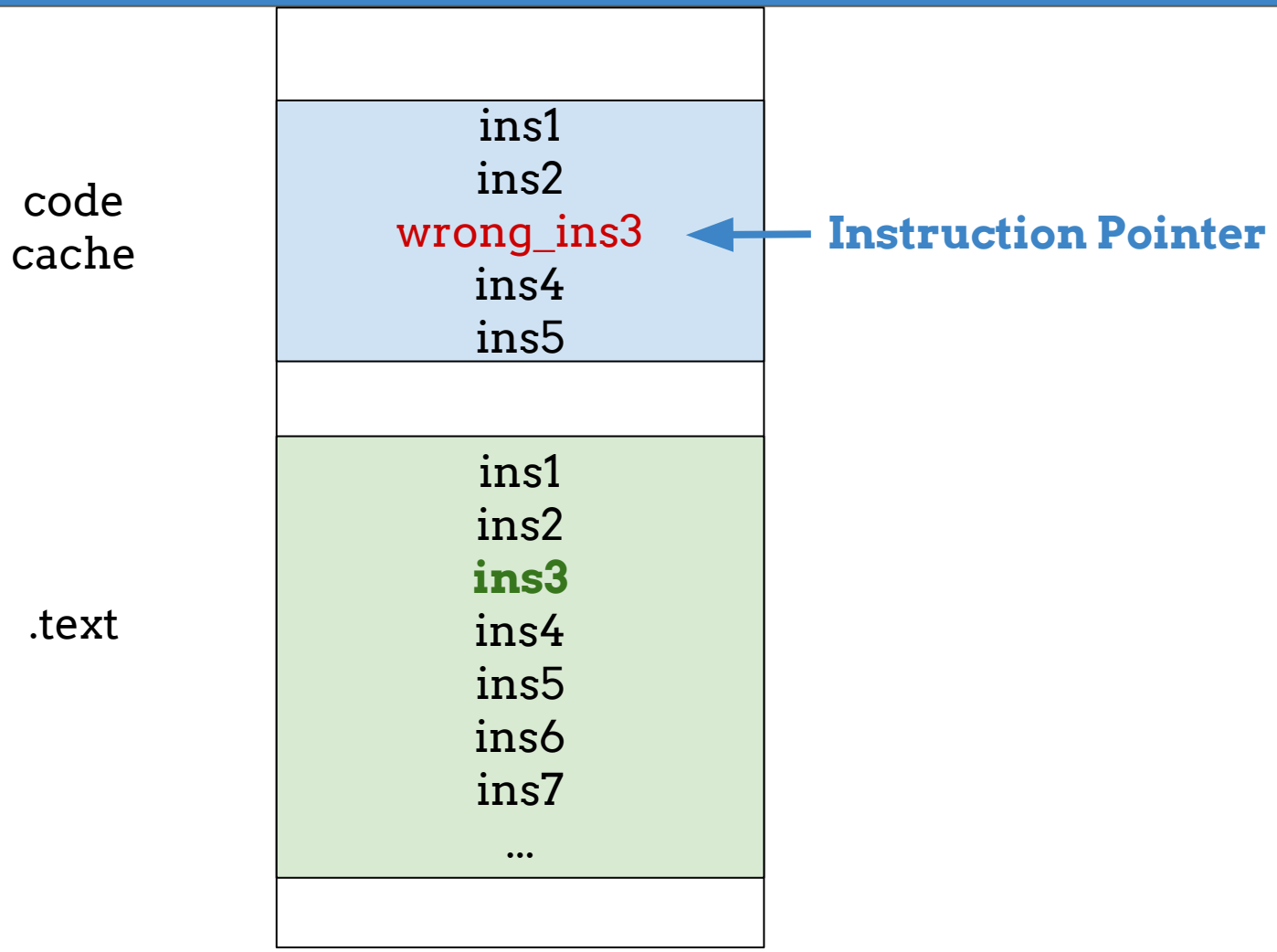
CCA - Self Modifying Code



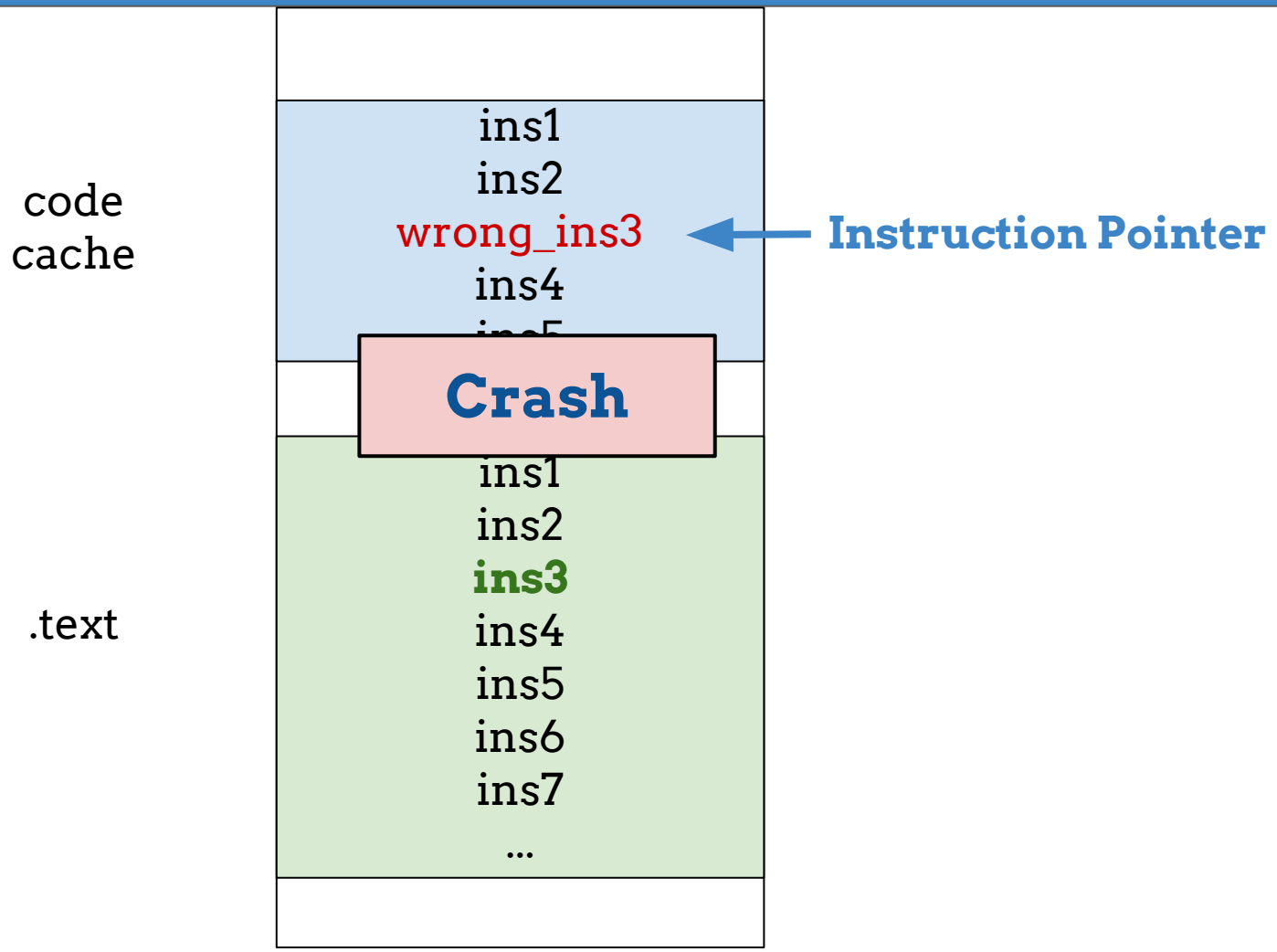
CCA - Self Modifying Code



CCA - Self Modifying Code



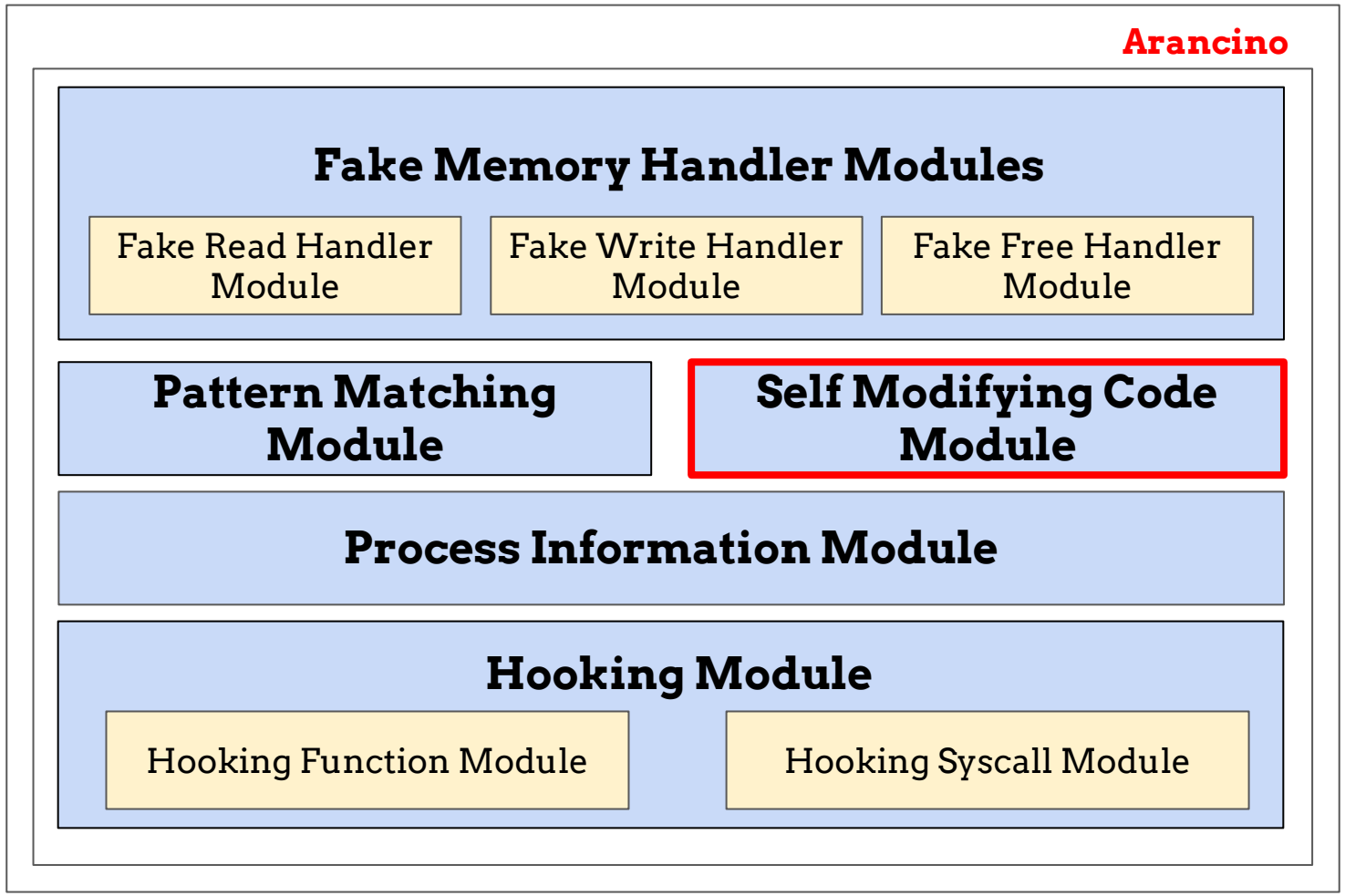
CCA - Self Modifying Code



Arancino - Self Modifying Code Module

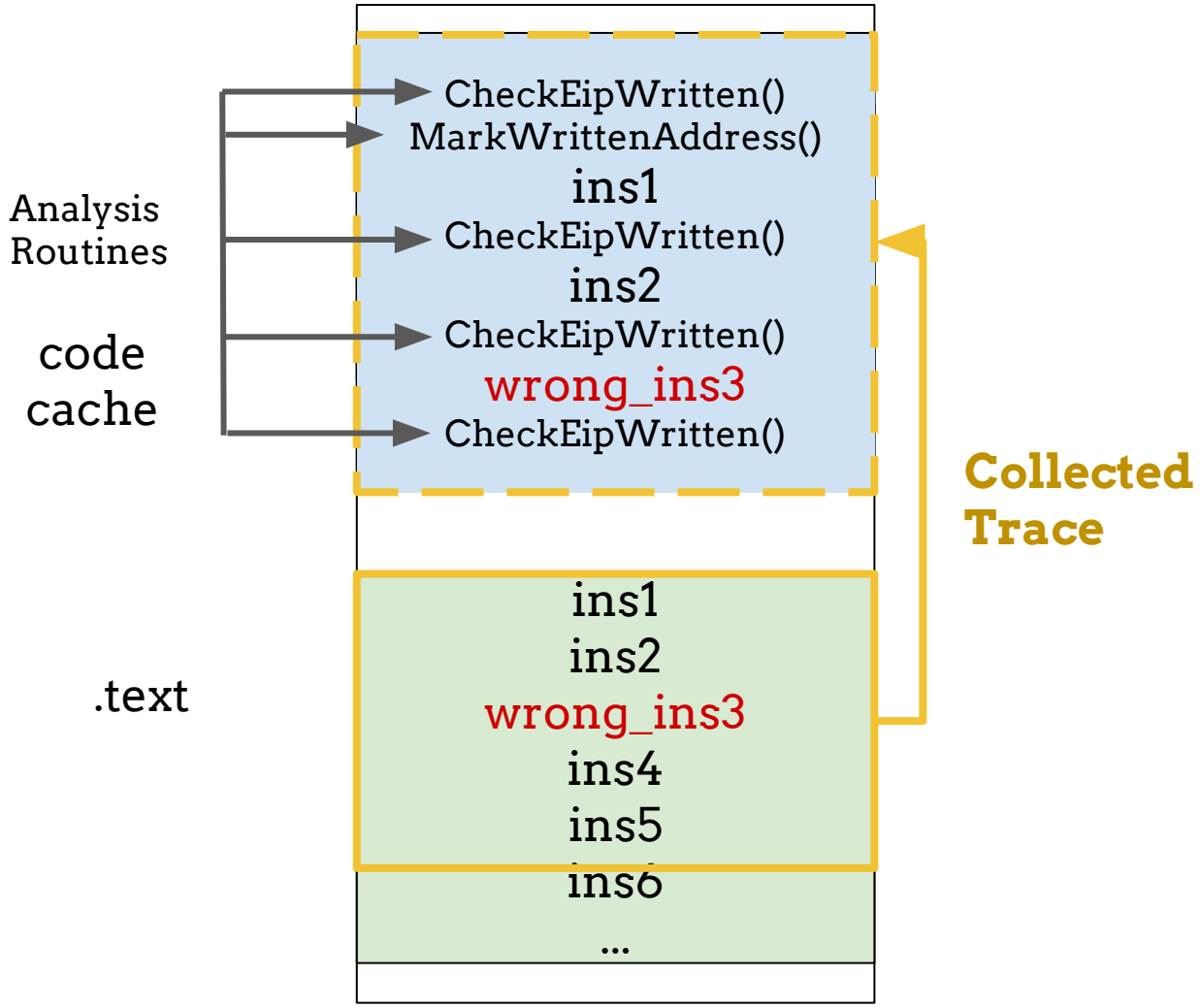
- **MarkWrittenAddress**: store which address has been overwritten

- **CheckEIPWritten**: check if next instruction has been overwritten.



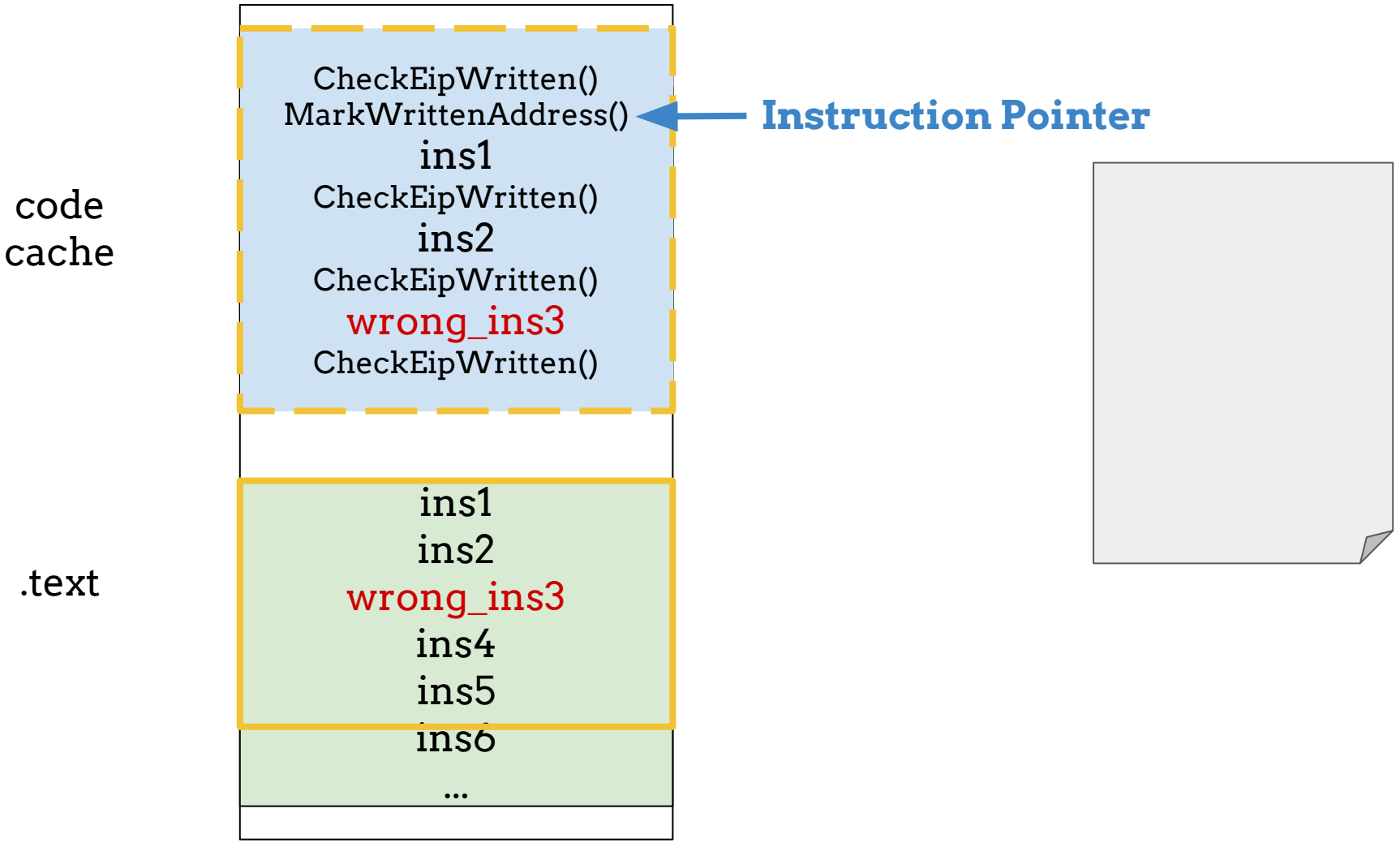
POLITECNICO MILANO 1863

CCA - Self Modifying Code



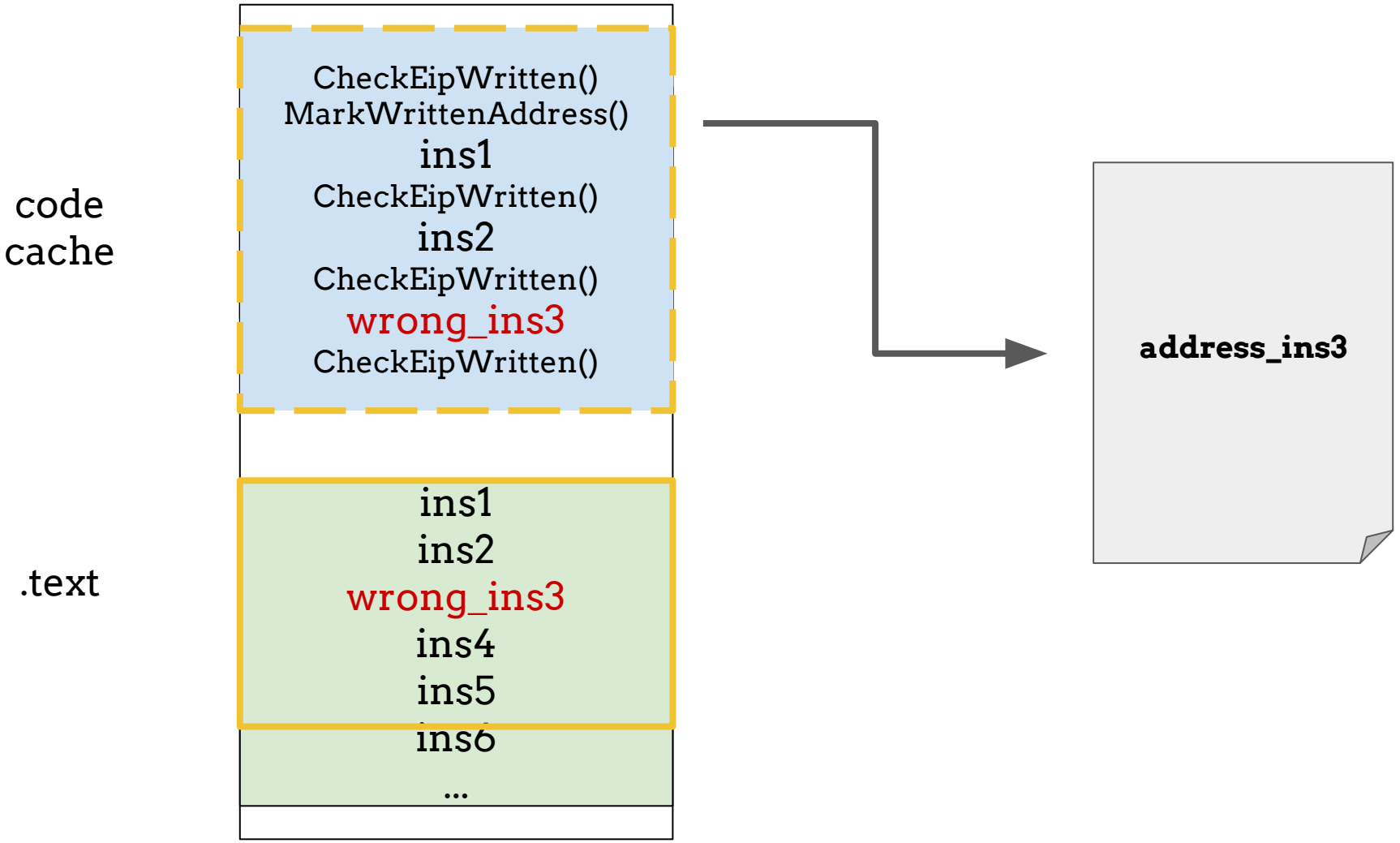
POLITECNICO MILANO 1863

CCA - Self Modifying Code



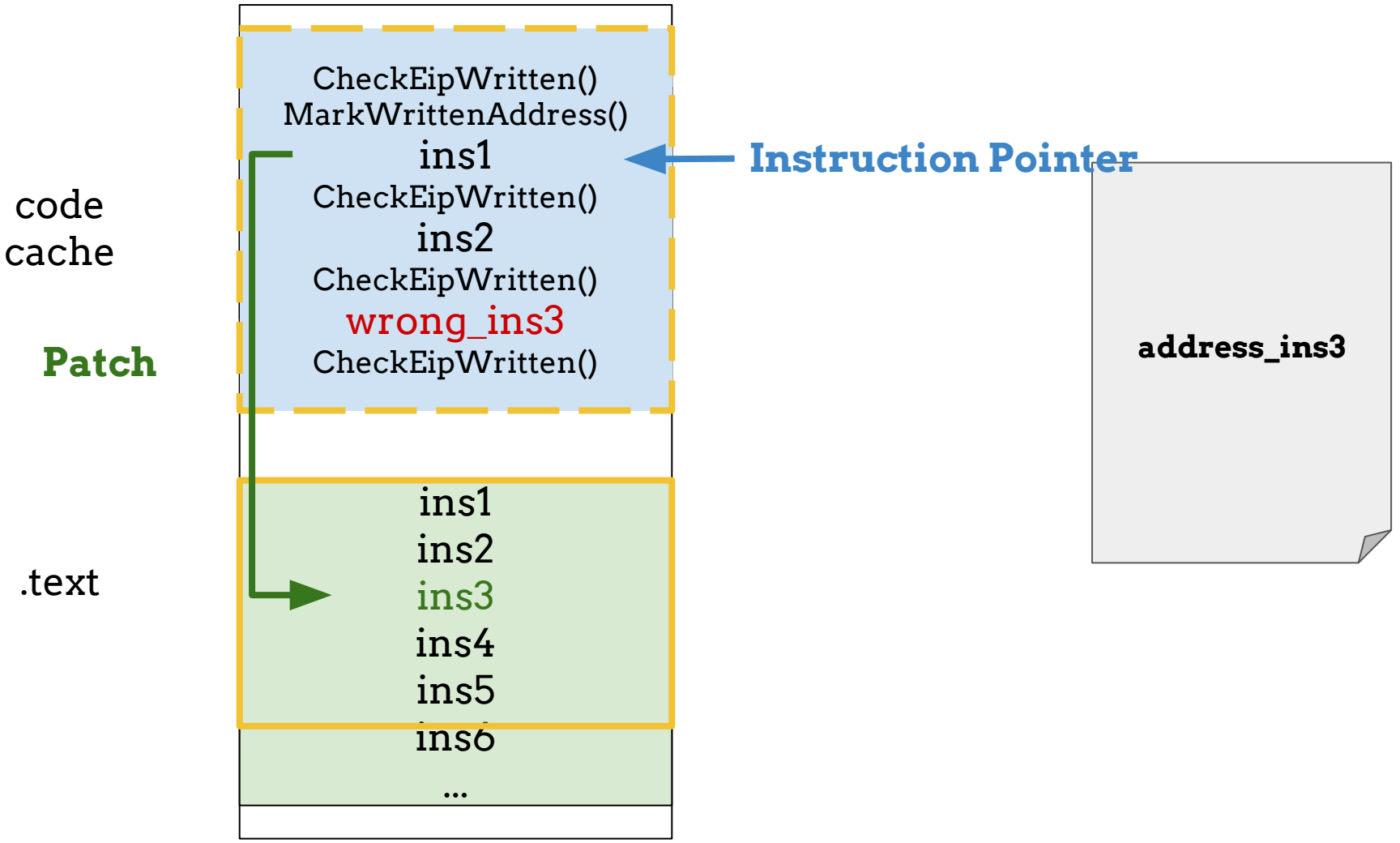
POLITECNICO MILANO 1863

CCA - Self Modifying Code



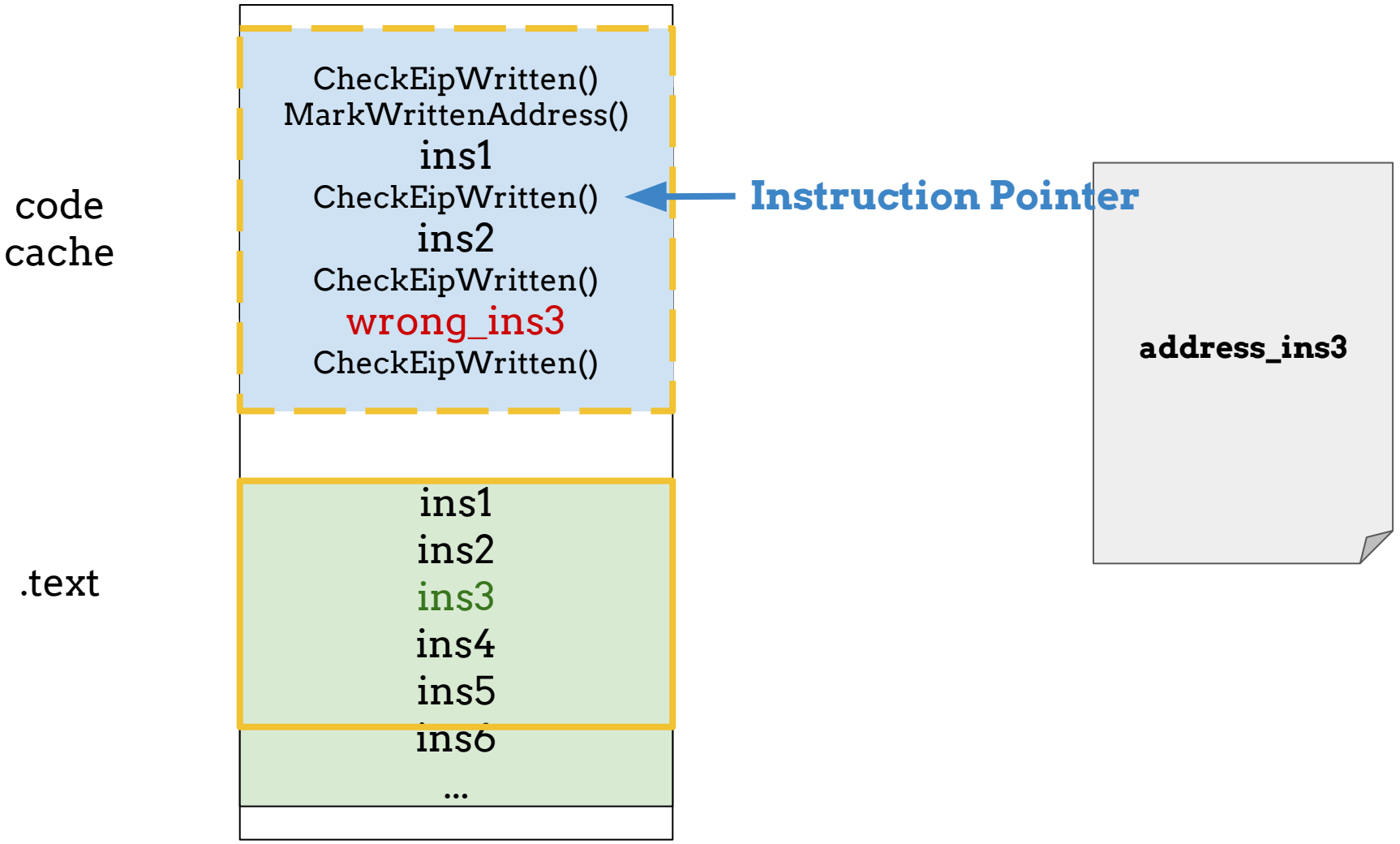
POLITECNICO MILANO 1863

CCA - Self Modifying Code



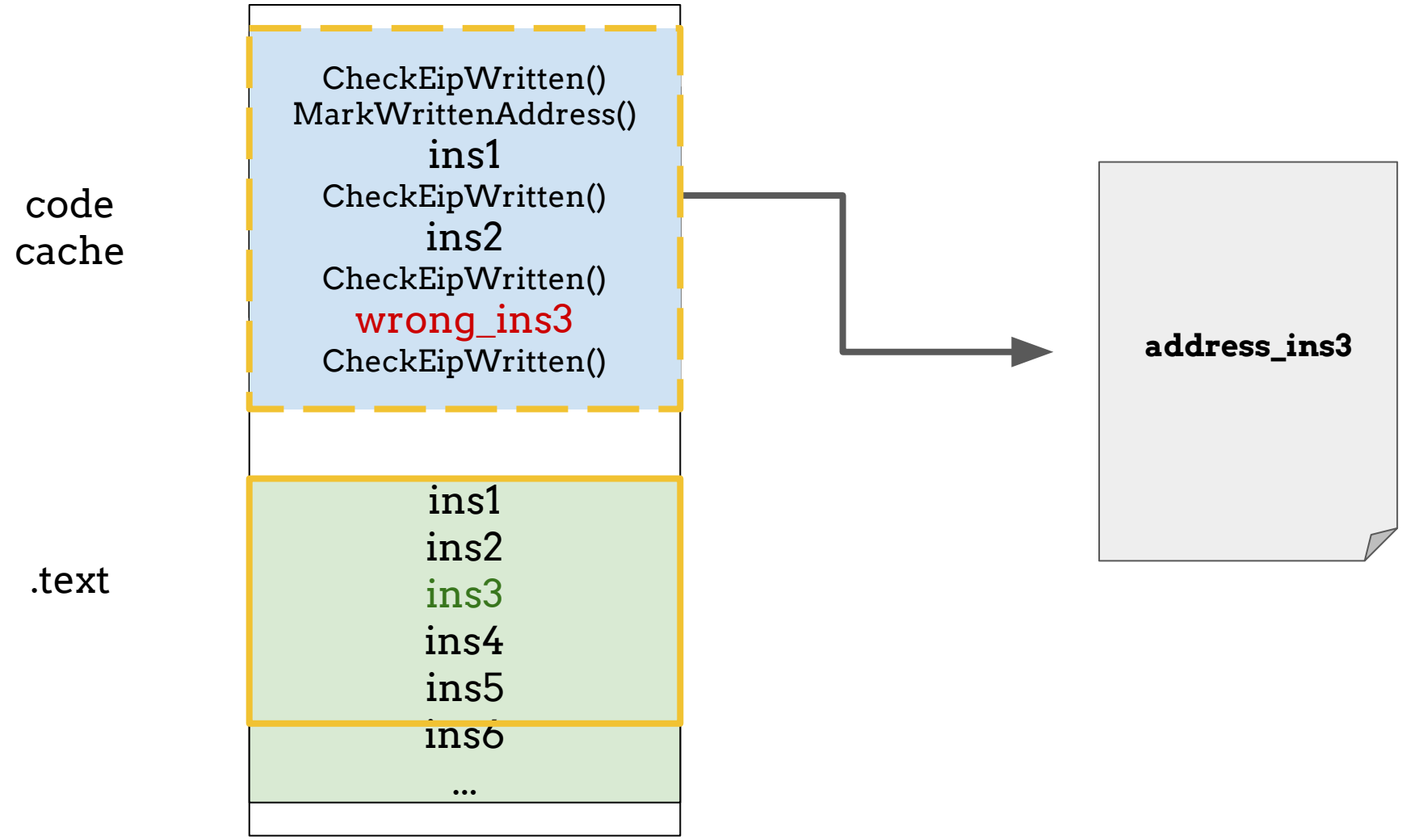
POLITECNICO MILANO 1863

CCA - Self Modifying Code



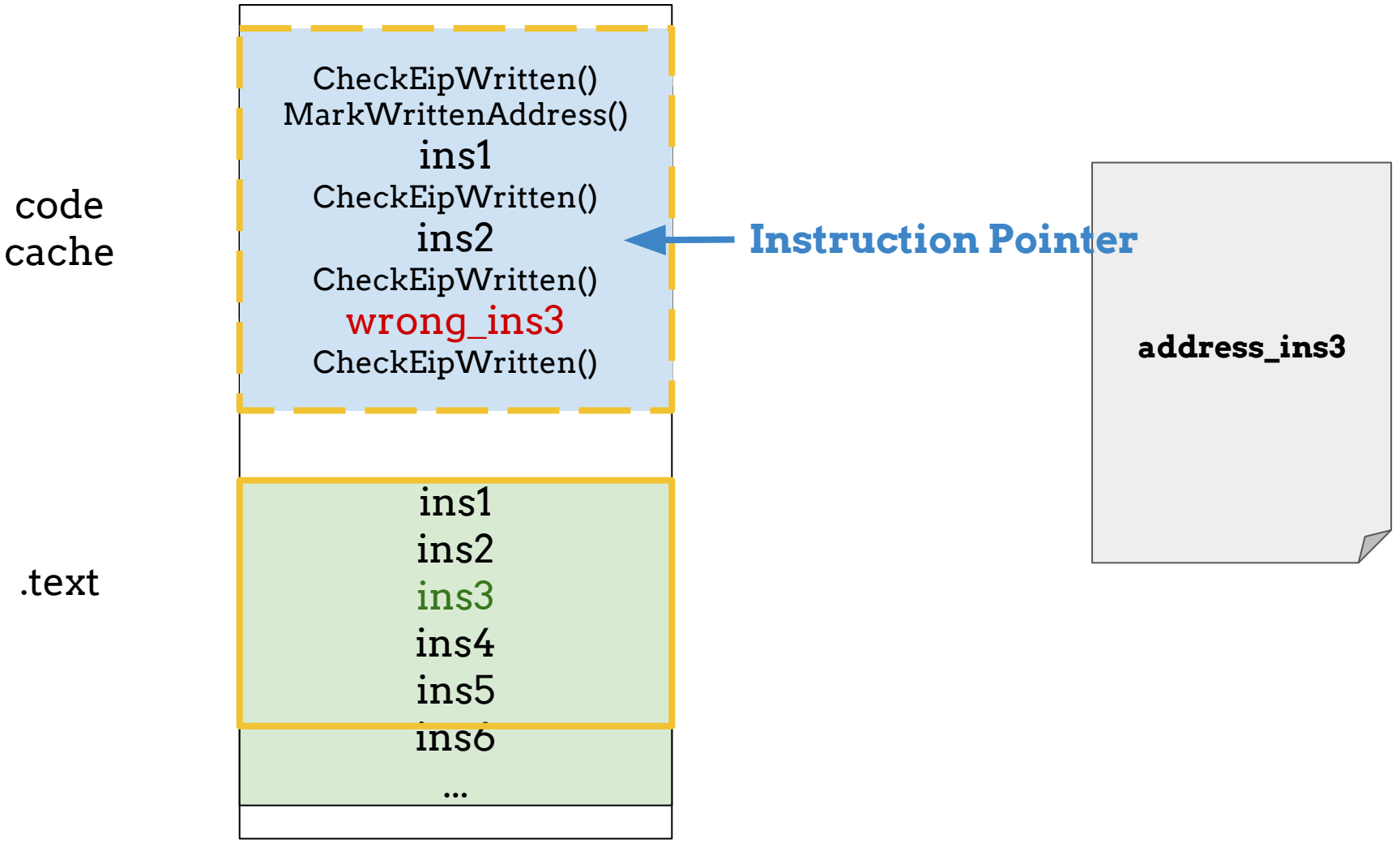
POLITECNICO MILANO 1863

CCA - Self Modifying Code



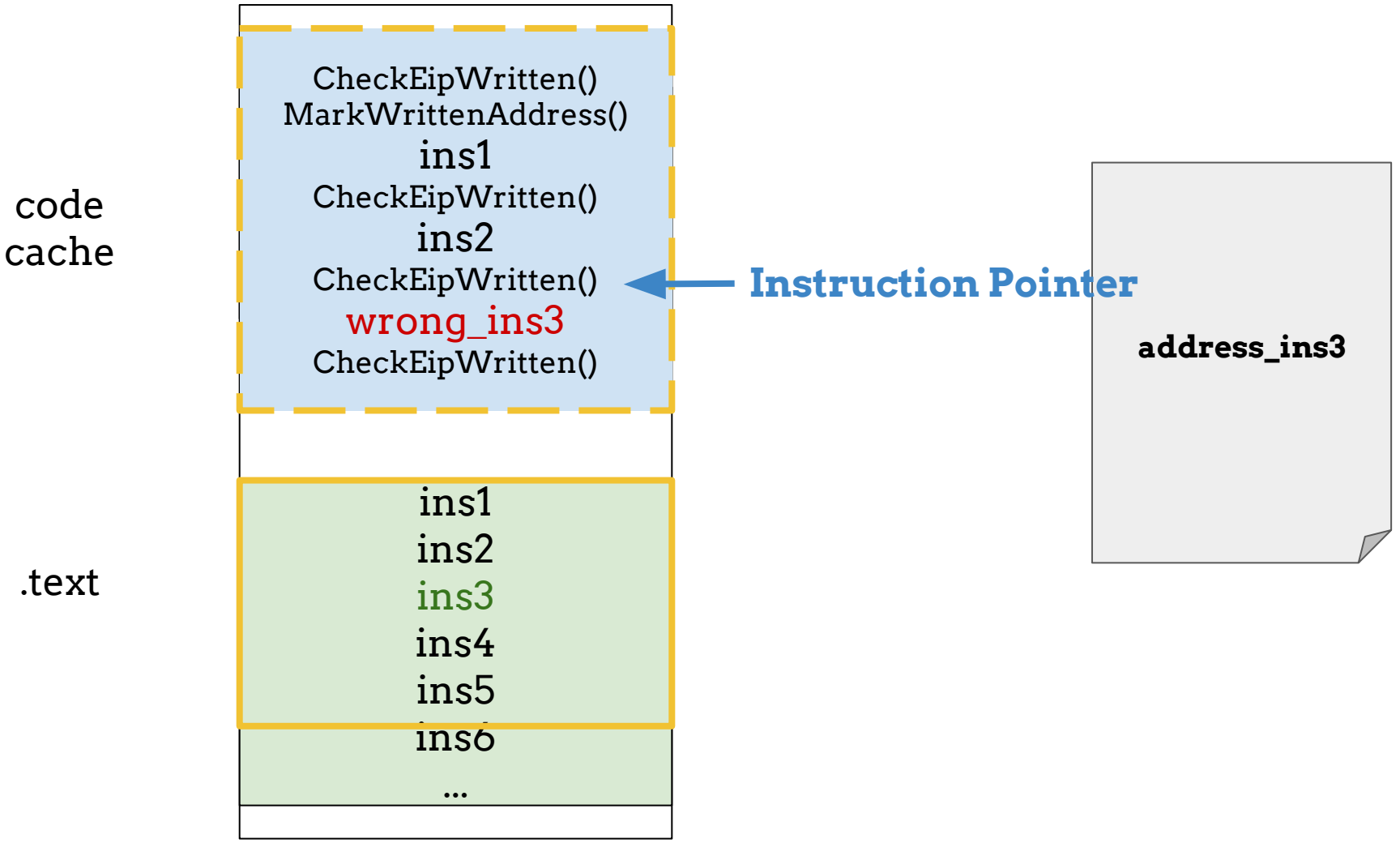
POLITECNICO MILANO 1863

CCA - Self Modifying Code



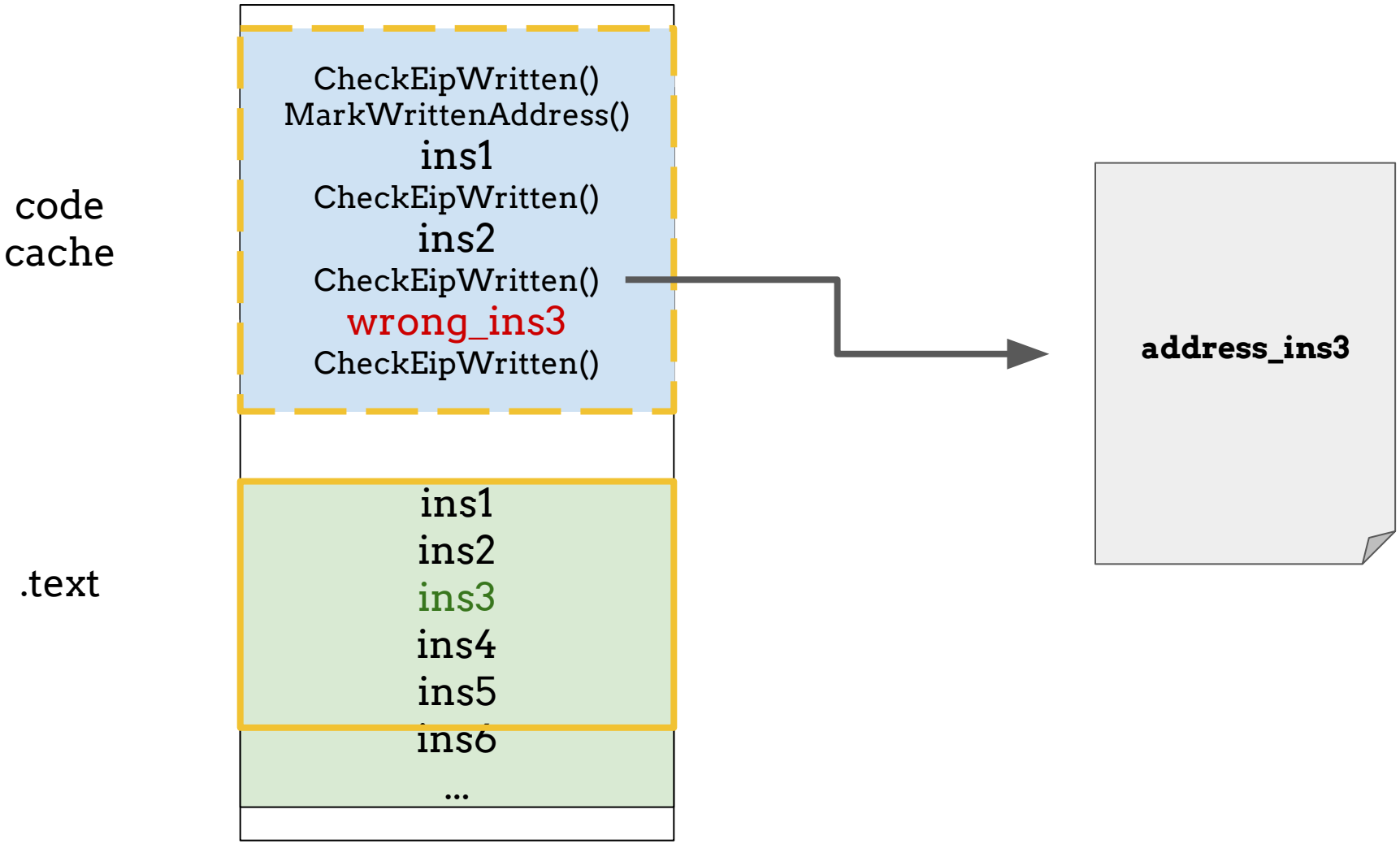
POLITECNICO MILANO 1863

CCA - Self Modifying Code



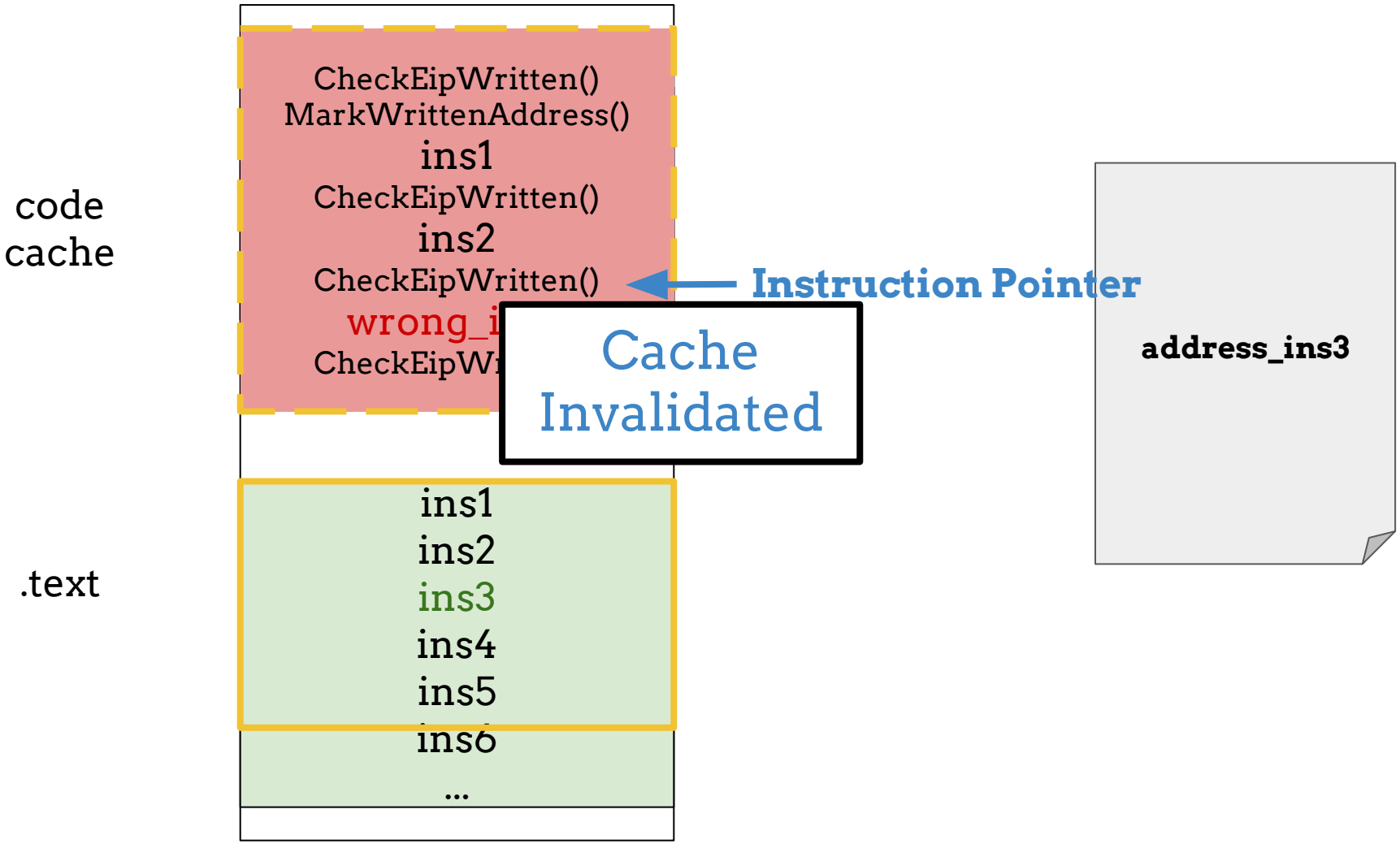
POLITECNICO MILANO 1863

CCA - Self Modifying Code



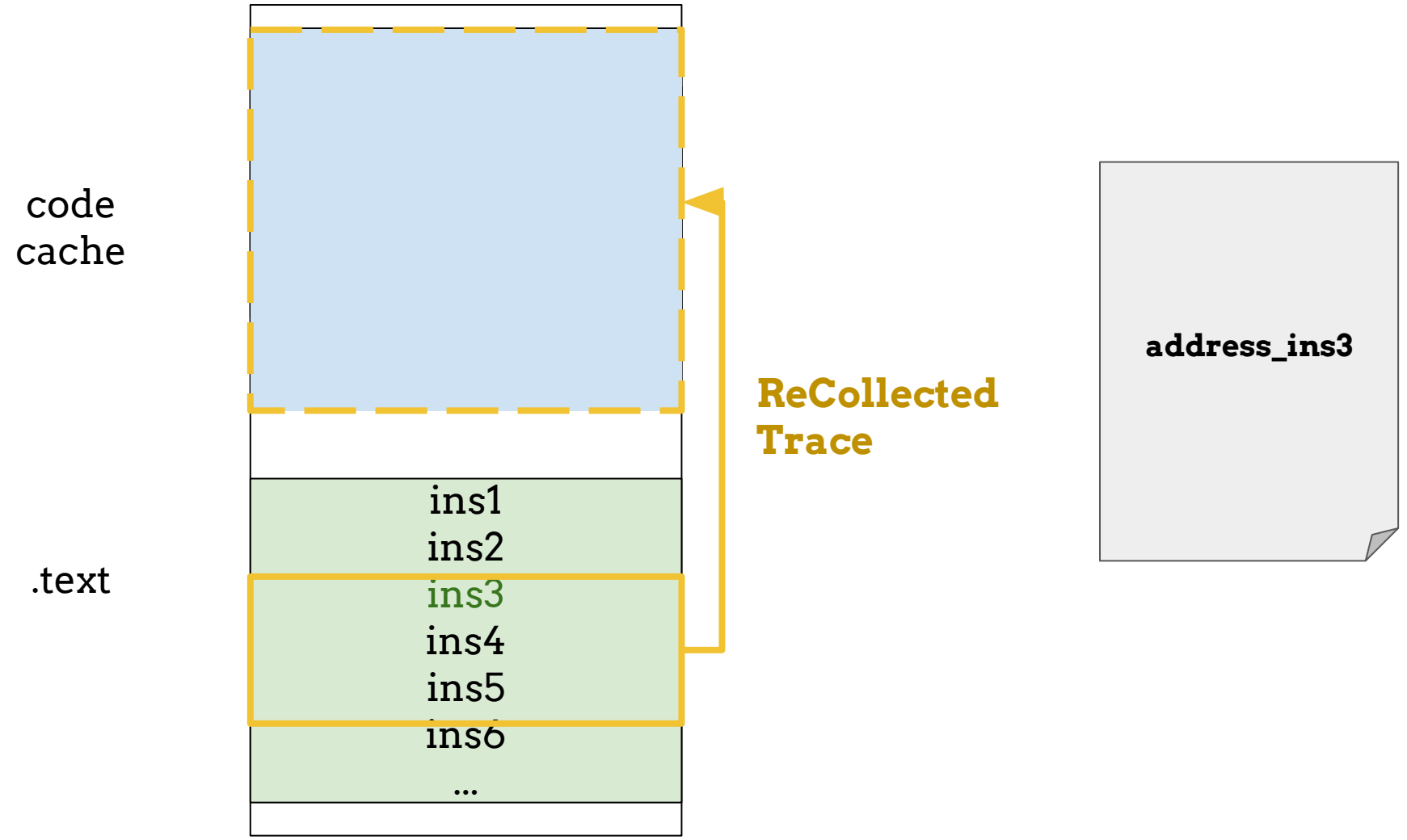
POLITECNICO MILANO 1863

CCA - Self Modifying Code



POLITECNICO MILANO 1863

CCA - Self Modifying Code

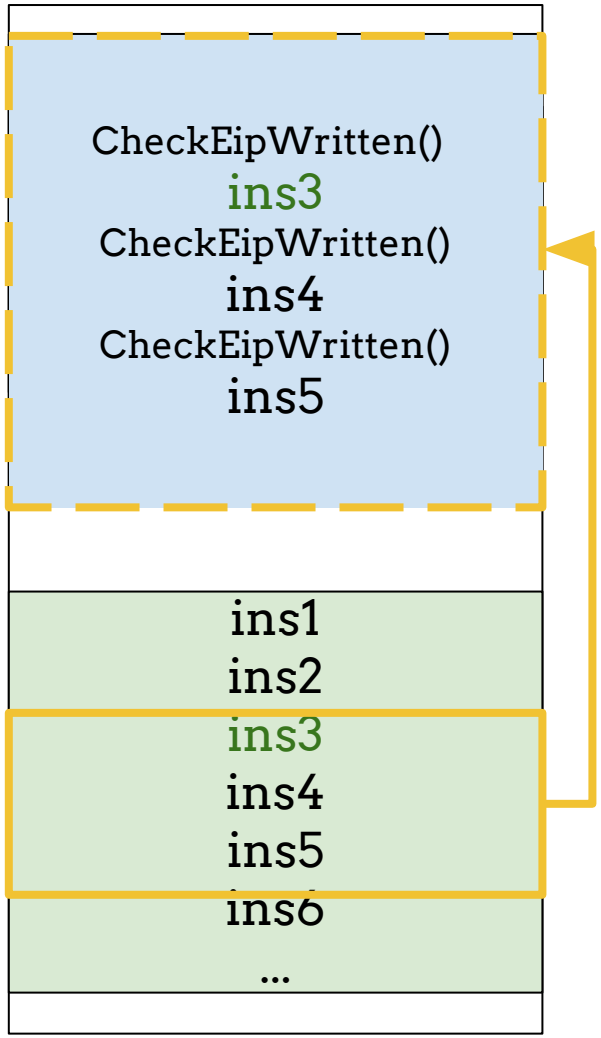


POLITECNICO MILANO 1863

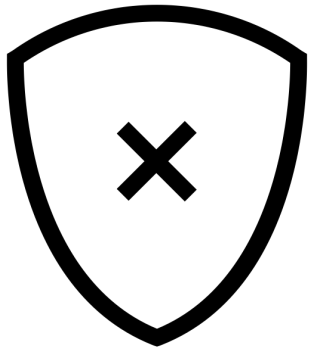
CCA - Self Modifying Code

code cache

.text



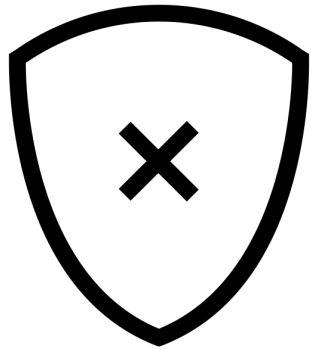
ReCollected Trace



Environment Artifacts

Environment Artifacts

- **Parent Detection**
- **Memory Fingerprinting**

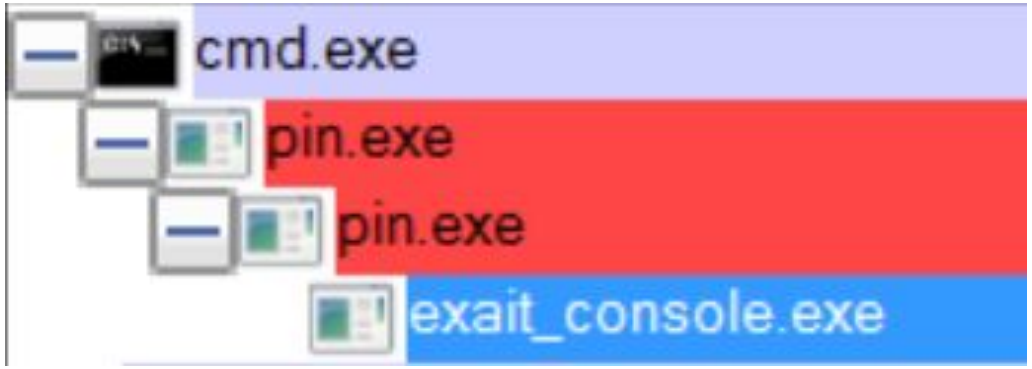


POLITECNICO
MILANO 1863

EA - Parent Detection

Malware can check which is the process father.

- NtQuerySystemInformation
- CSRSS.exe

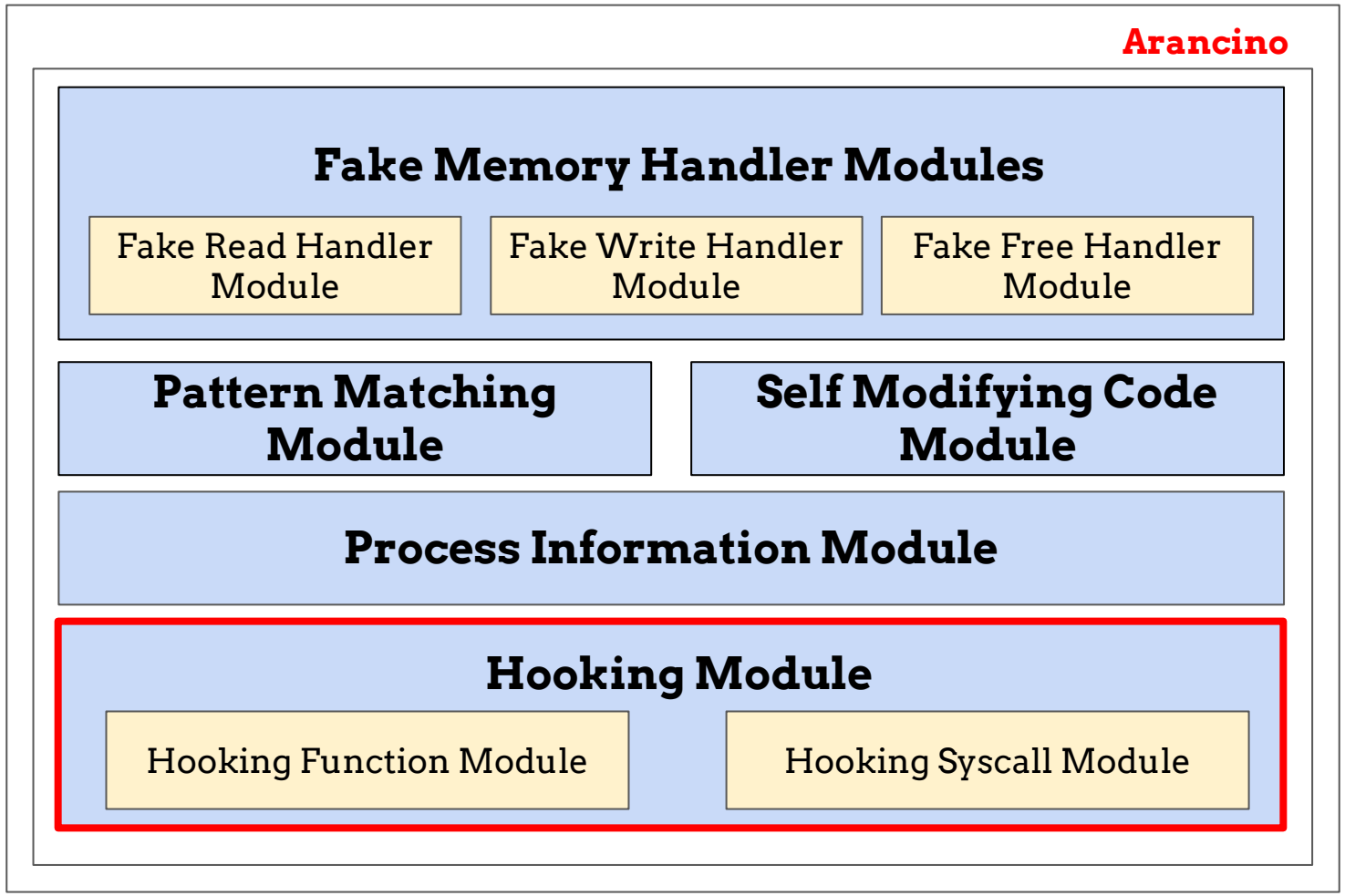


POLITECNICO MILANO 1863

Arancino - Hooking Module

- **Hooking Function Module**: Install an Hook on dll's Functions

- **Hooking Syscall Module**: Install an Hook on dll's Functions

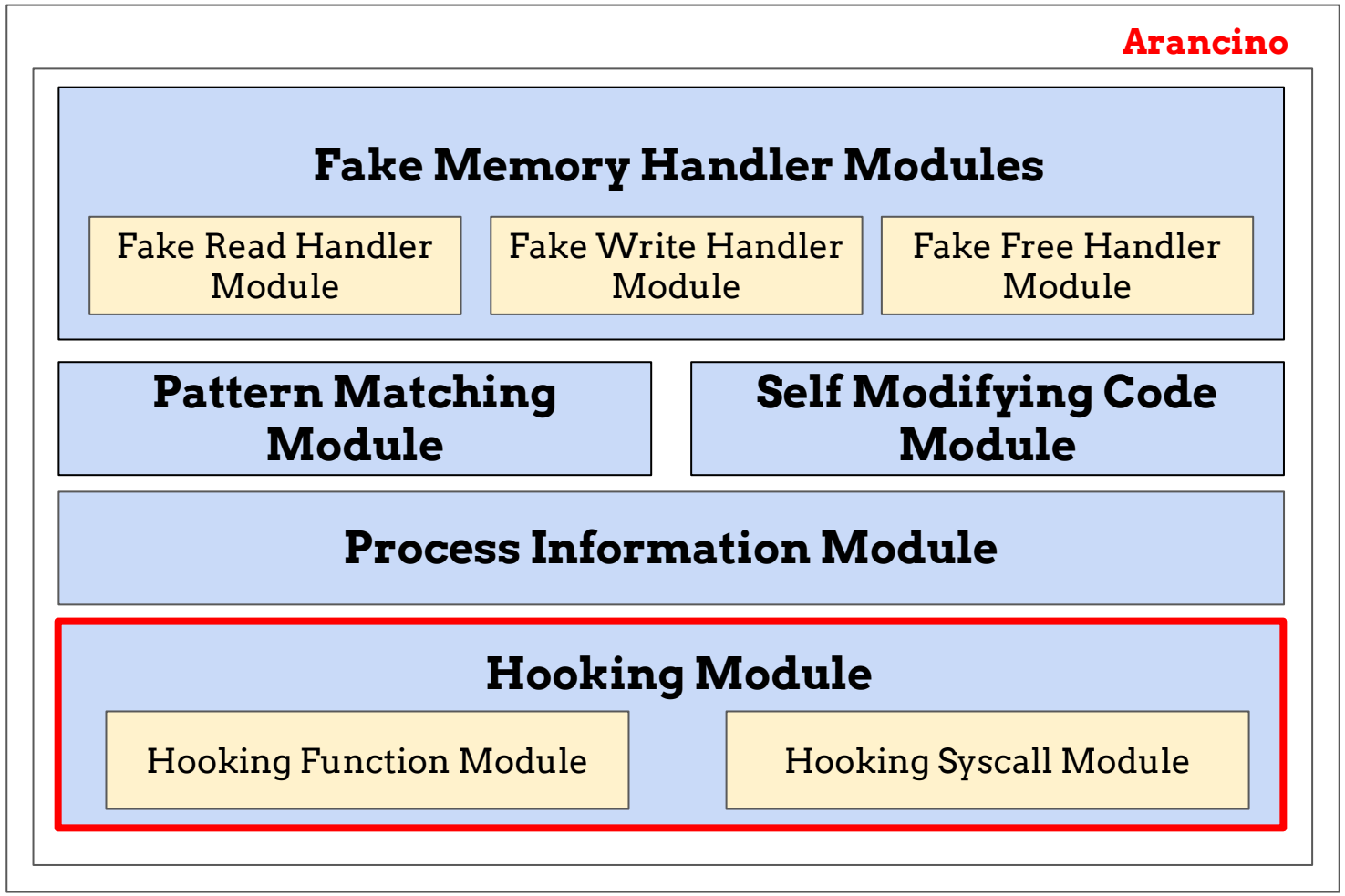


POLITECNICO MILANO 1863

Arancino - Hooking Module

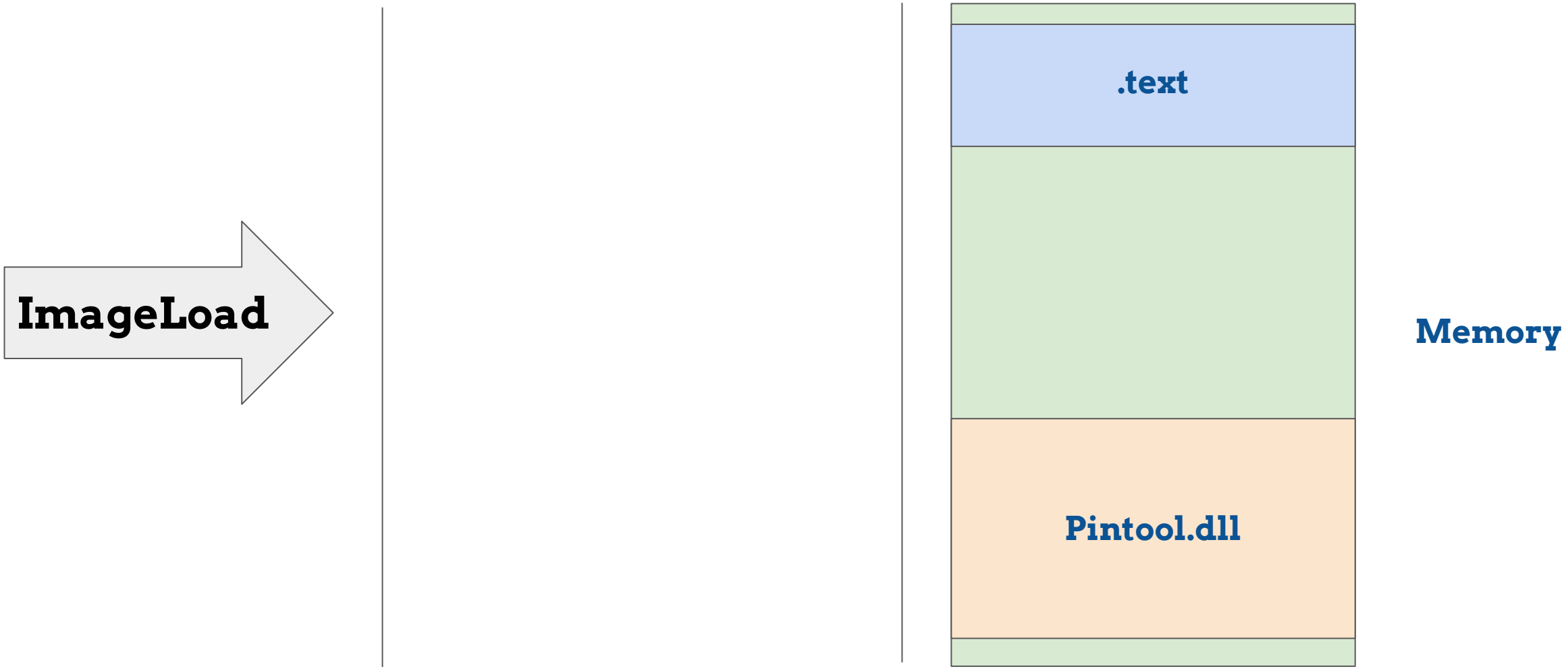
- **Hooking Function Module**: Install an Hook on dll's Functions

- **Hooking Syscall Module**: Install an Hook on dll's Functions



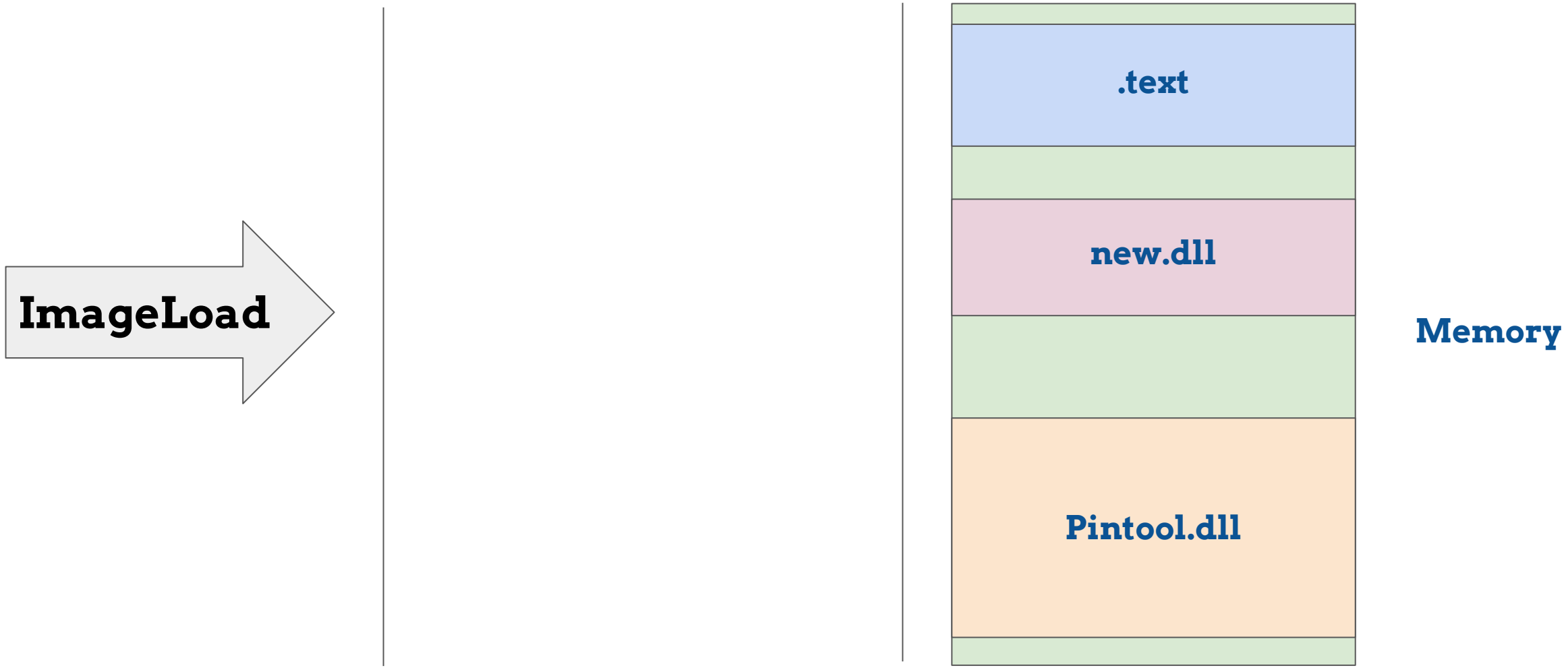
POLITECNICO MILANO 1863

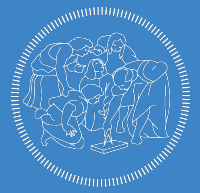
Arancino - Hook Functions



POLITECNICO
MILANO 1863

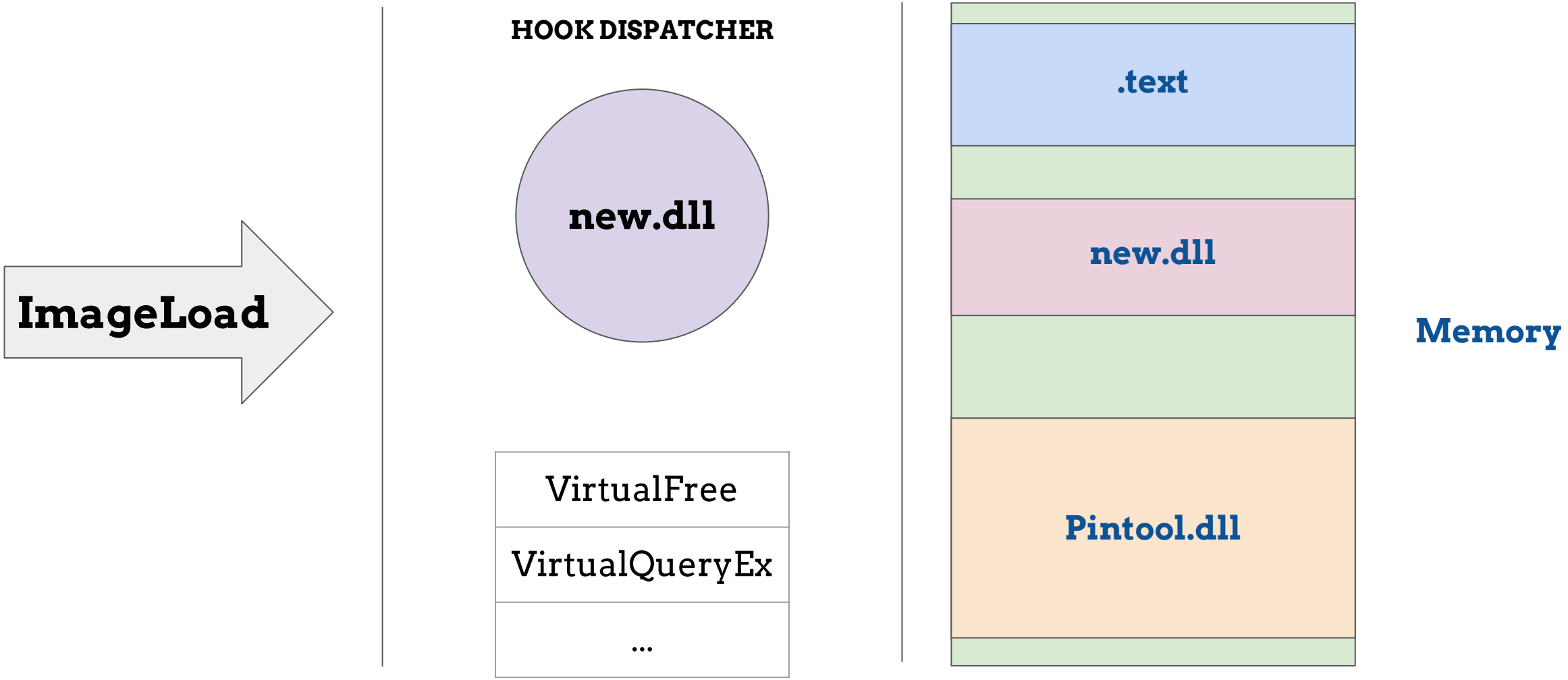
Arancino - Hook Functions

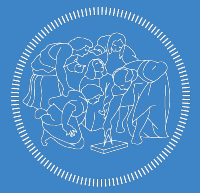




POLITECNICO
MILANO 1863

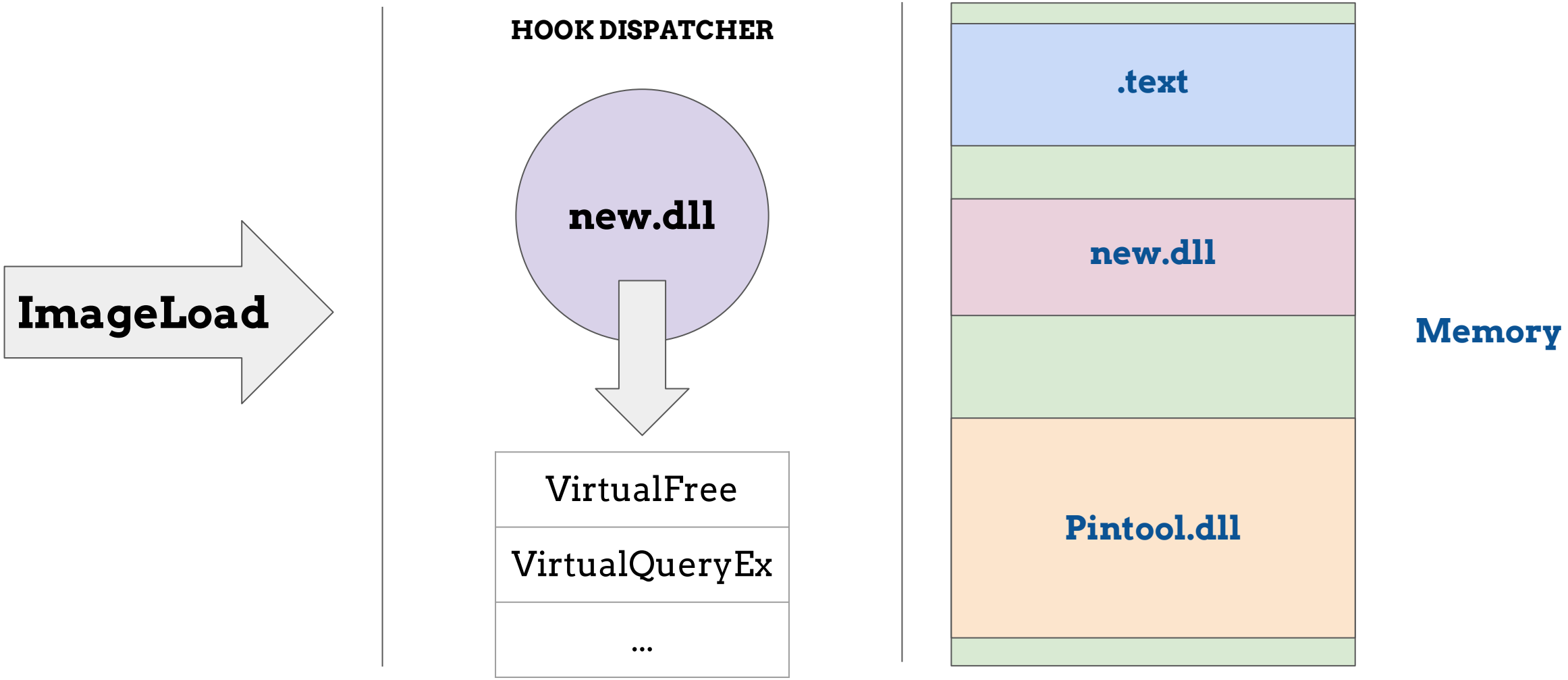
Arancino - Hook Functions





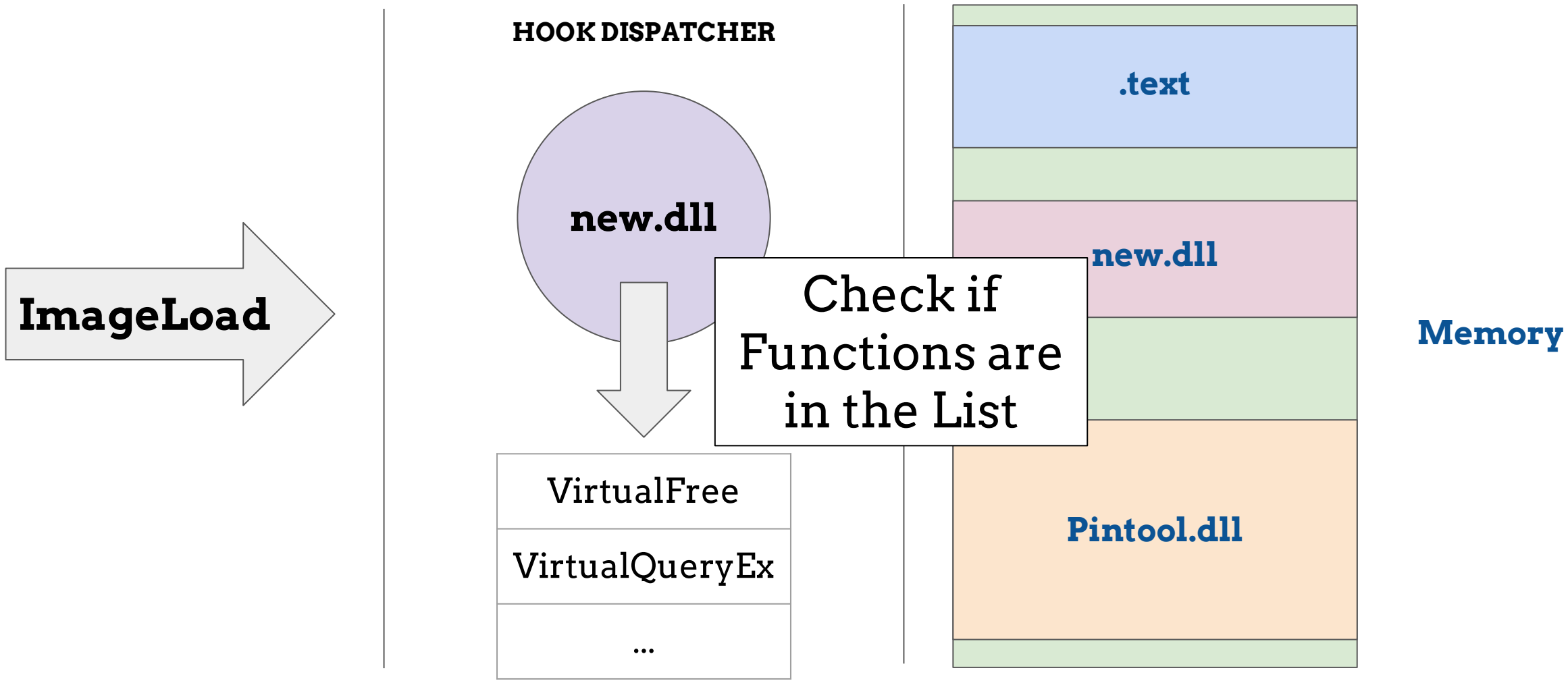
POLITECNICO
MILANO 1863

Arancino - Hook Functions



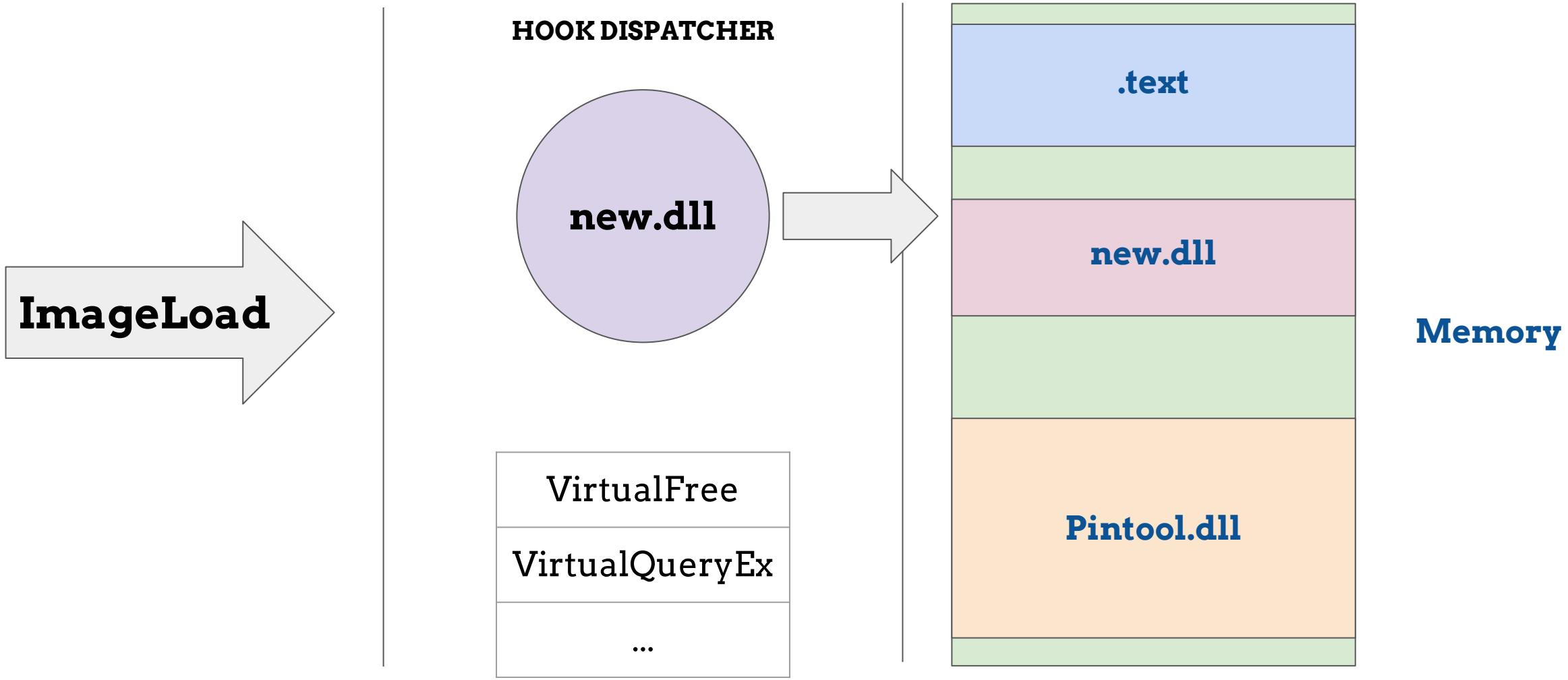
POLITECNICO MILANO 1863

Arancino - Hook Functions



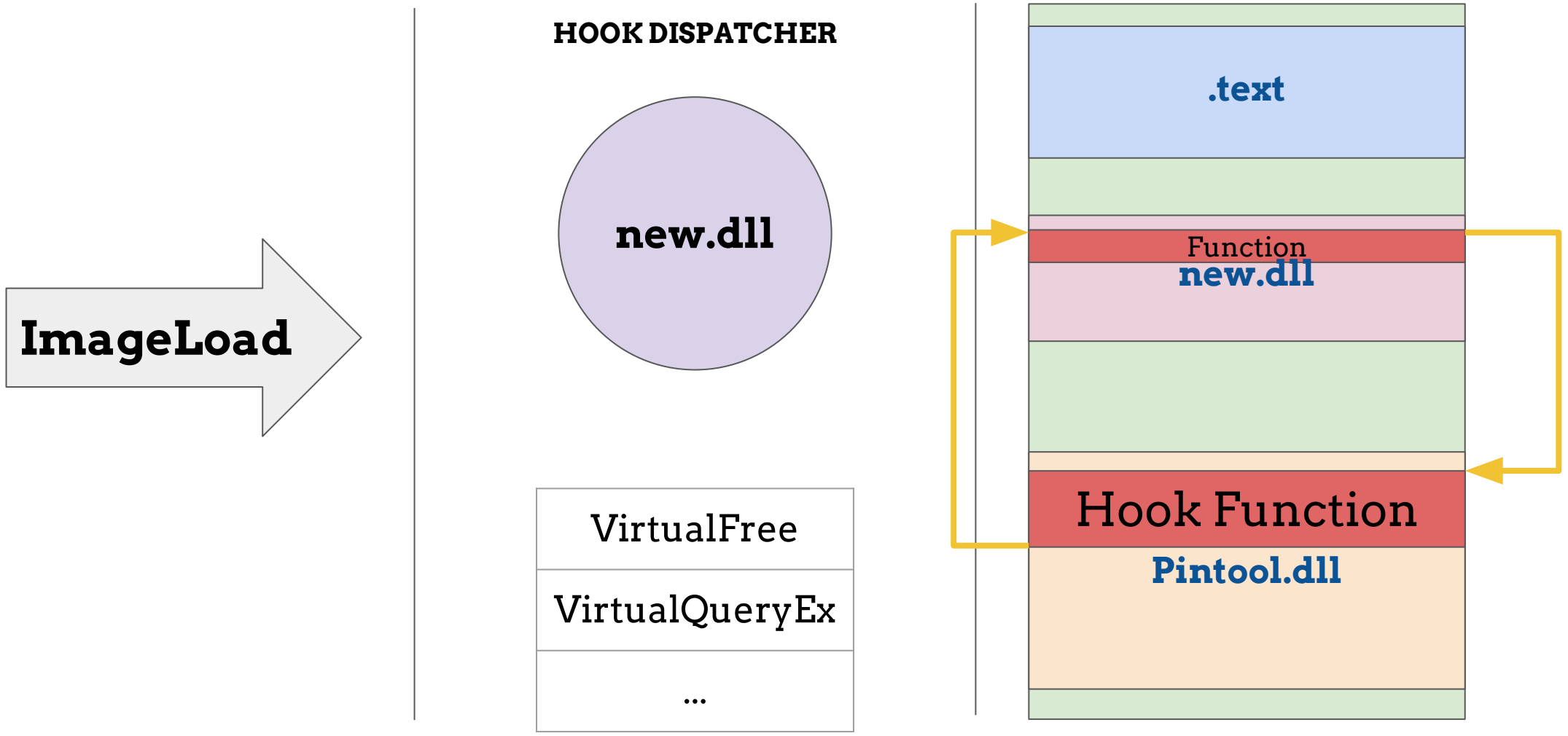
POLITECNICO MILANO 1863

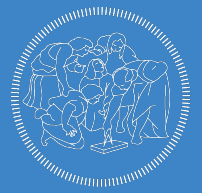
Arancino - Hook Functions



POLITECNICO MILANO 1863

Arancino - Hook Functions





POLITECNICO
MILANO 1863

EA - Parent Detection

Hooked NtQuerySystemInformation

`pin.exe -> cmd.exe`

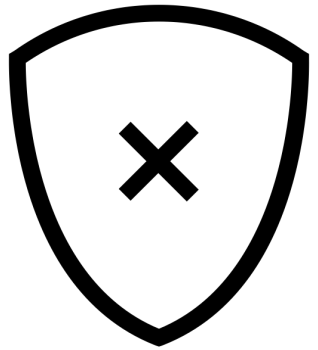
Hooked NtOpenProcess

to deny access to `CSRSS.exe`

POLITECNICO
MILANO 1863

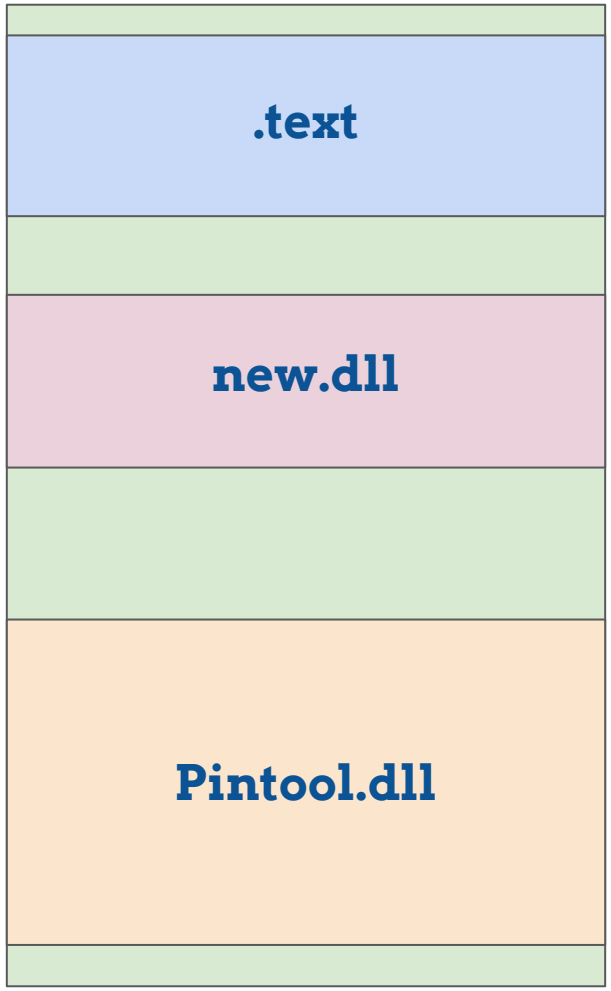
Environment Artifacts

- **Parent Detection**

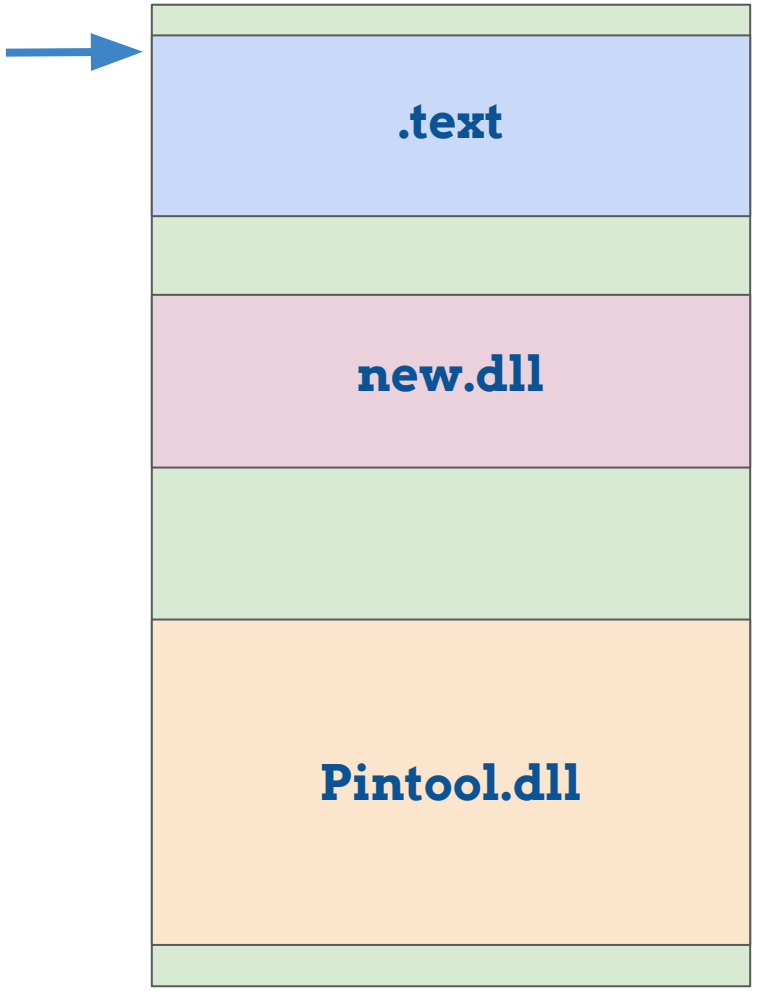


- **Memory Fingerprinting**

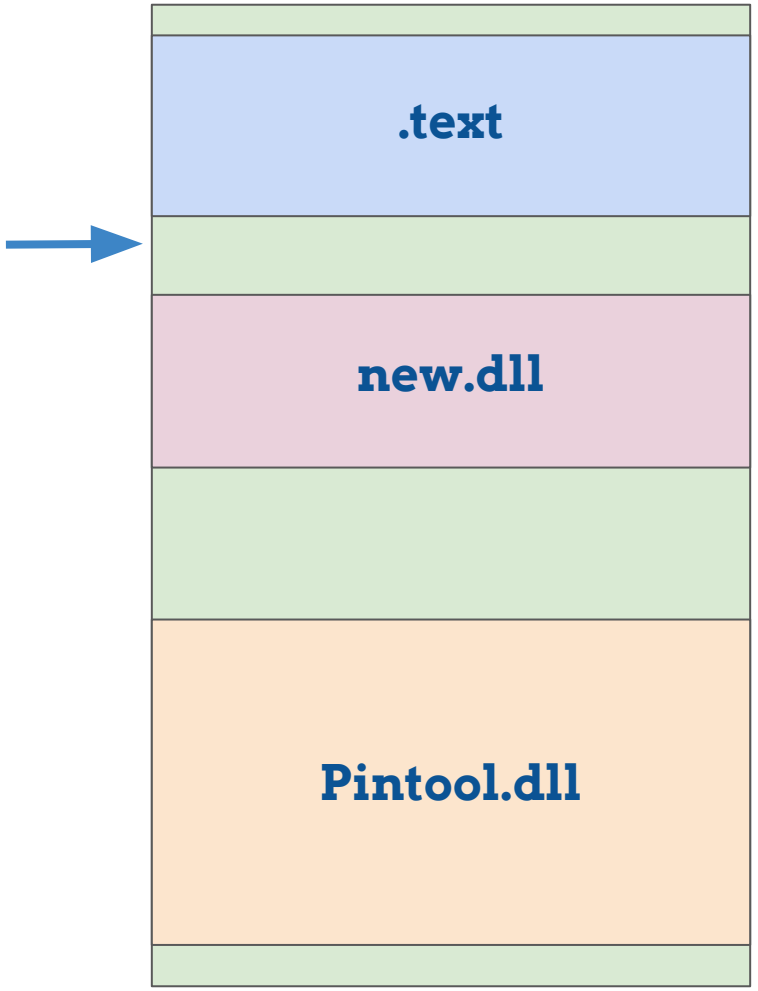
EA - Memory Fingerprinting



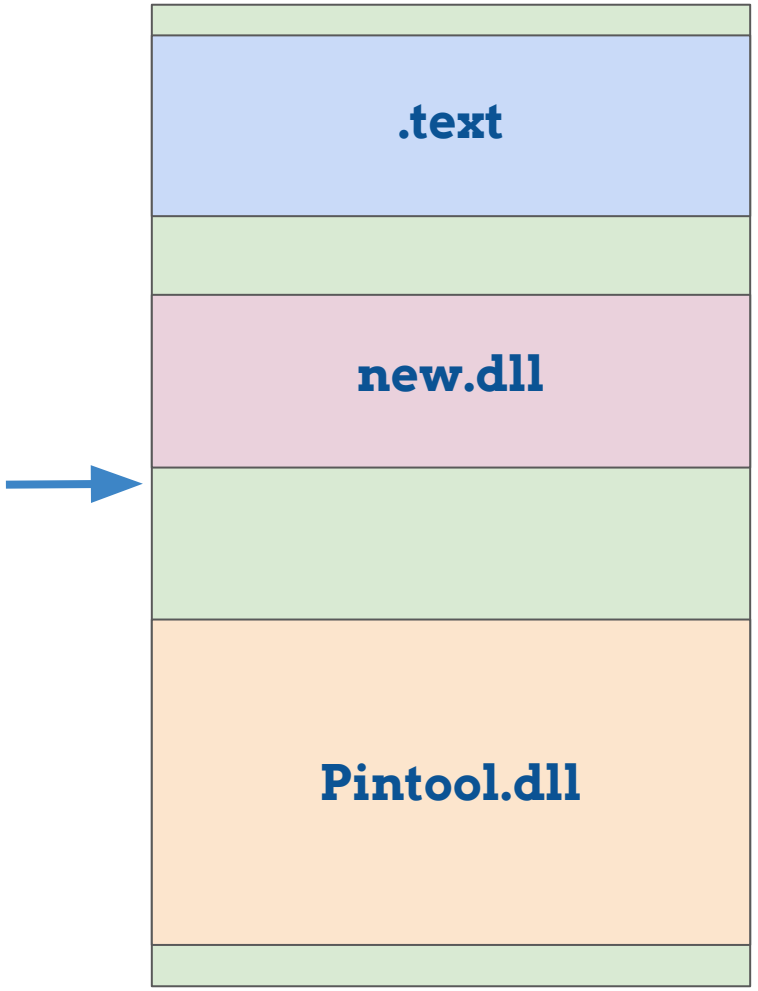
EA - Memory Fingerprinting



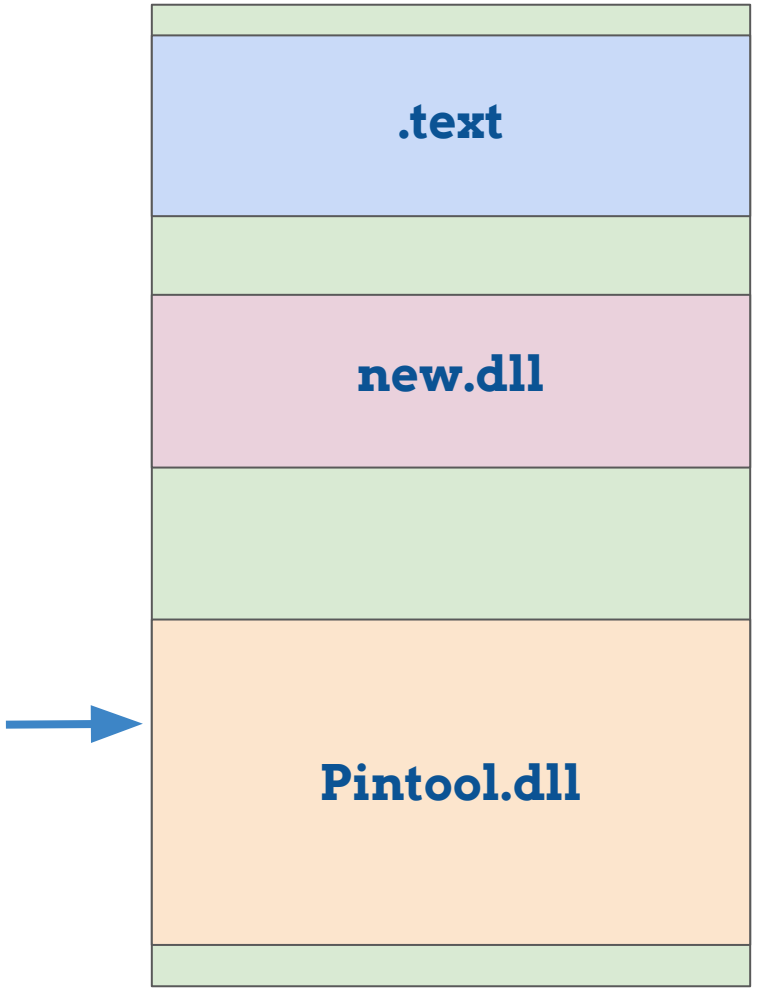
EA - Memory Fingerprinting



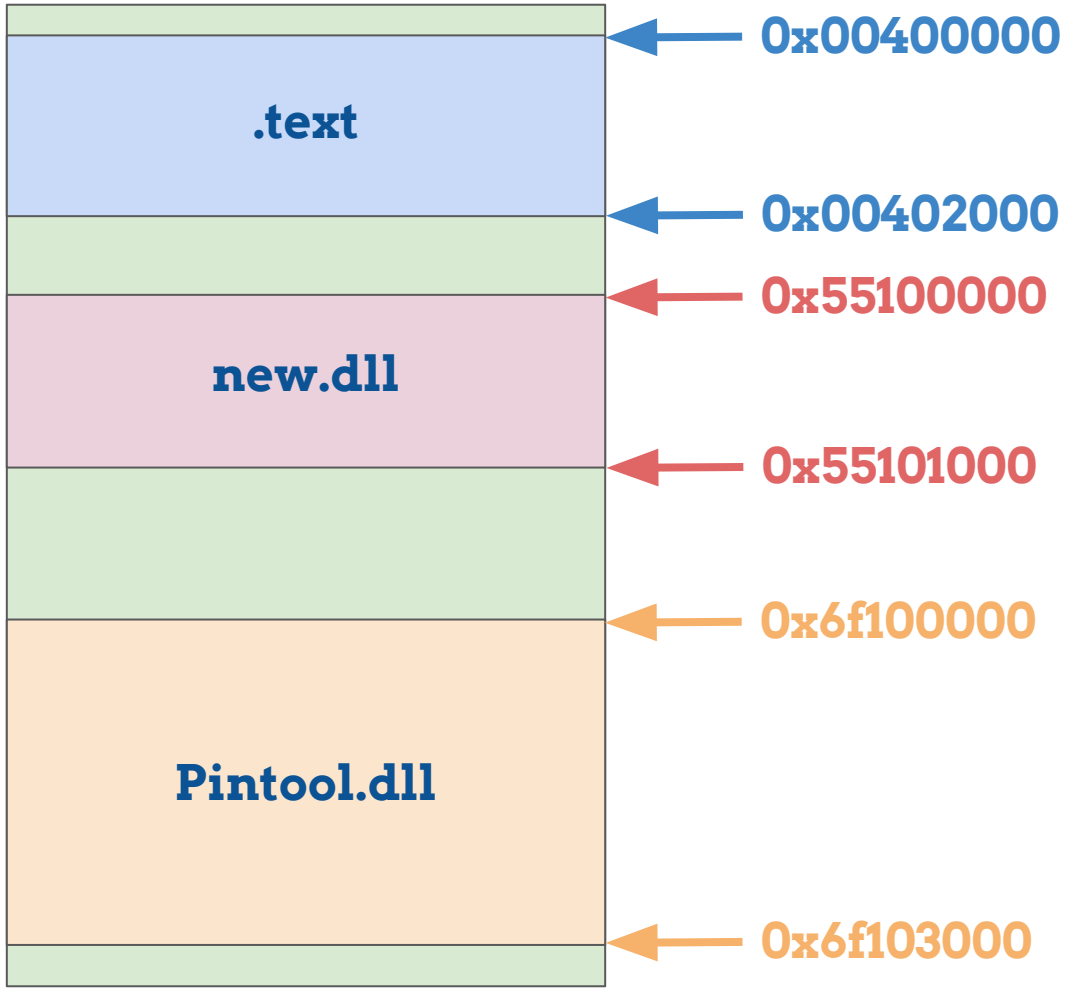
EA - Memory Fingerprinting



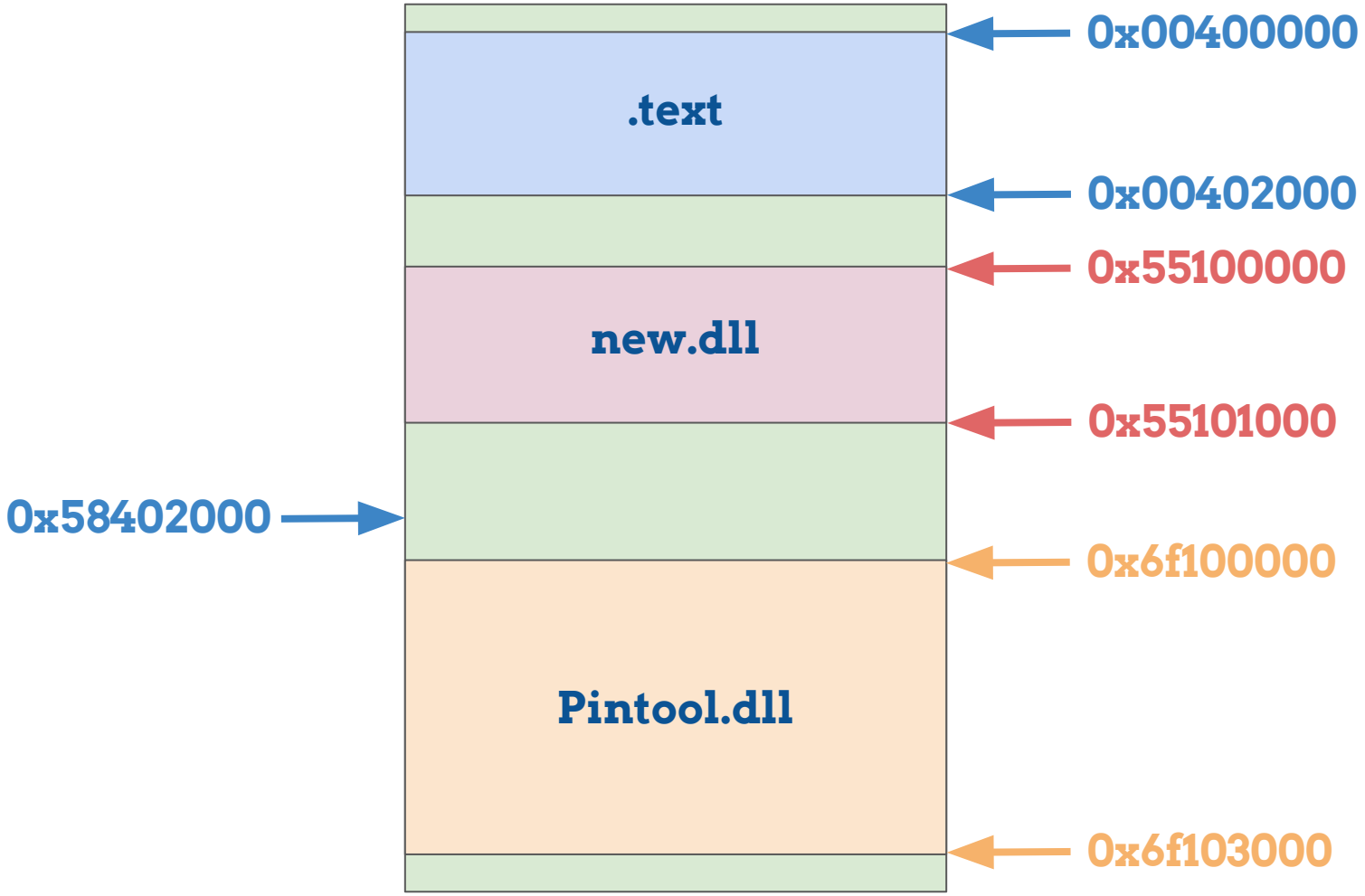
EA - Memory Fingerprinting



EA - Memory Fingerprinting

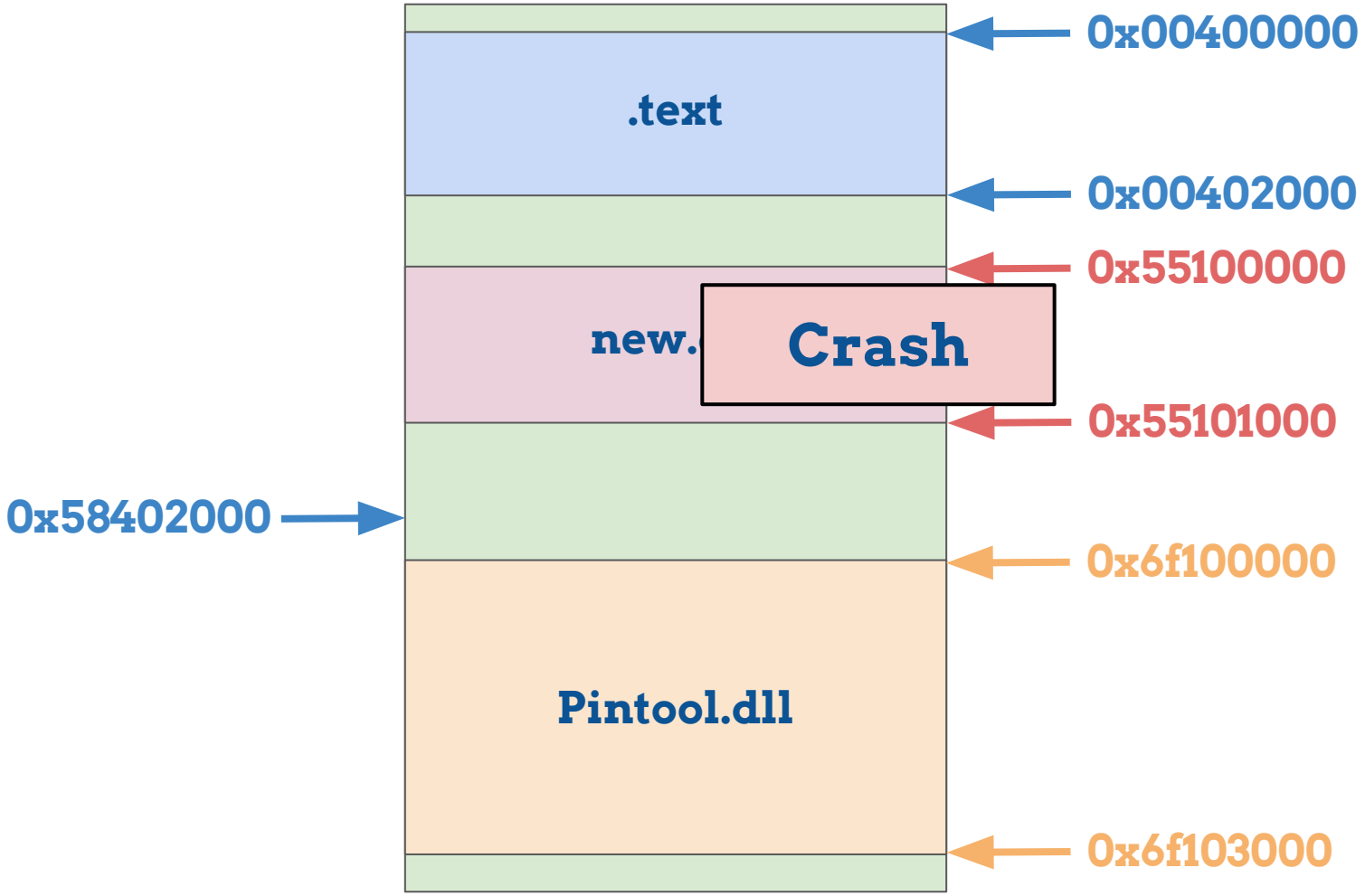


EA - Memory Fingerprinting

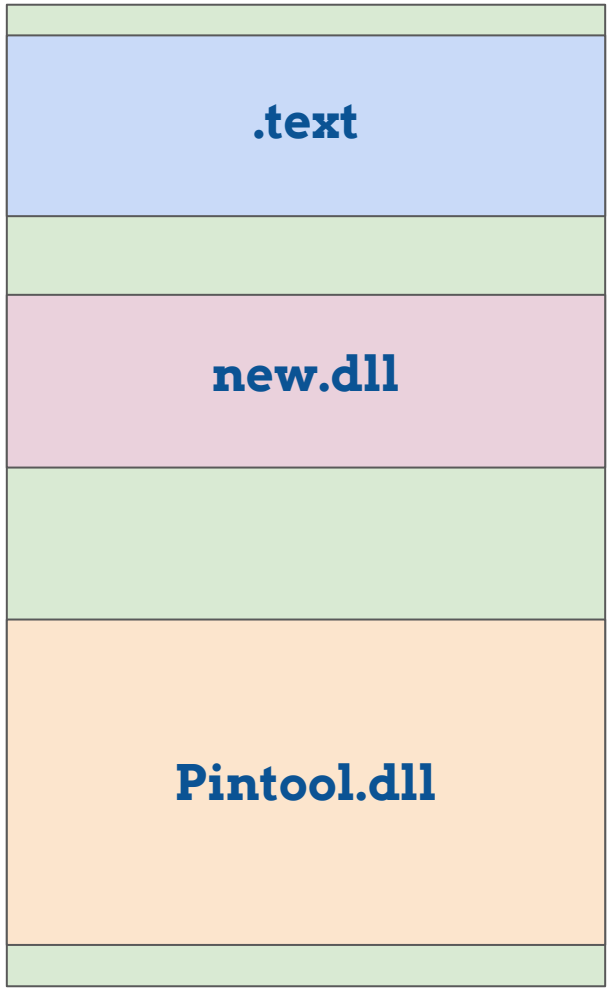


POLITECNICO MILANO 1863

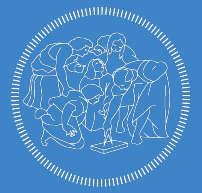
EA - Memory Fingerprinting



EA - Memory Fingerprinting



VirtualQuery



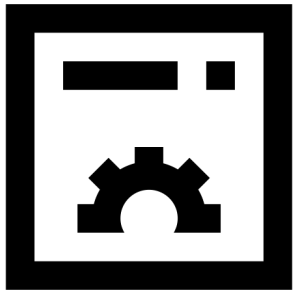
POLITECNICO
MILANO 1863

EA - Memory Fingerprinting

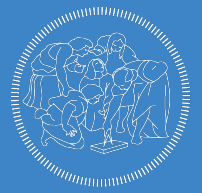
We Hook **NtQueryVirtualMemory**

We create a **Whitelist** of accessible memory regions updated at runtime.

- **Main Module**
- **Libraries**
- **Heap and Stack**
- **PEB, TEB, etc.**
- **Mapped files**



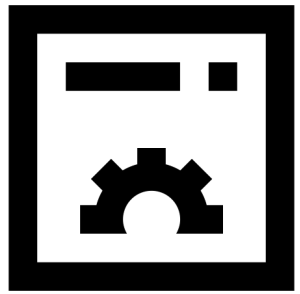
JIT Compiler Detection



POLITECNICO
MILANO 1863

JIT Compiler Detection

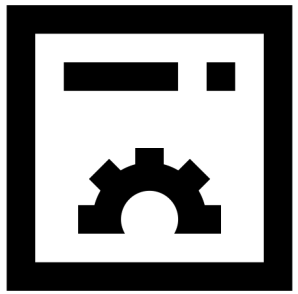
- **Memory Page Permissions**
  - Checks if there are **WX pages**
- **DLL Hook Detection**
- **Memory Allocations**



POLITECNICO
MILANO 1863

JIT Compiler Detection

- **Memory Page Permissions**
  - Checks if there are **WX pages**



- **DLL Hook Detection**

- **Memory Allocations**

POLITECNICO
MILANO 1863

JITC Detection - DLL Hook

A process can search through memory for discrepancy caused by Hooks.

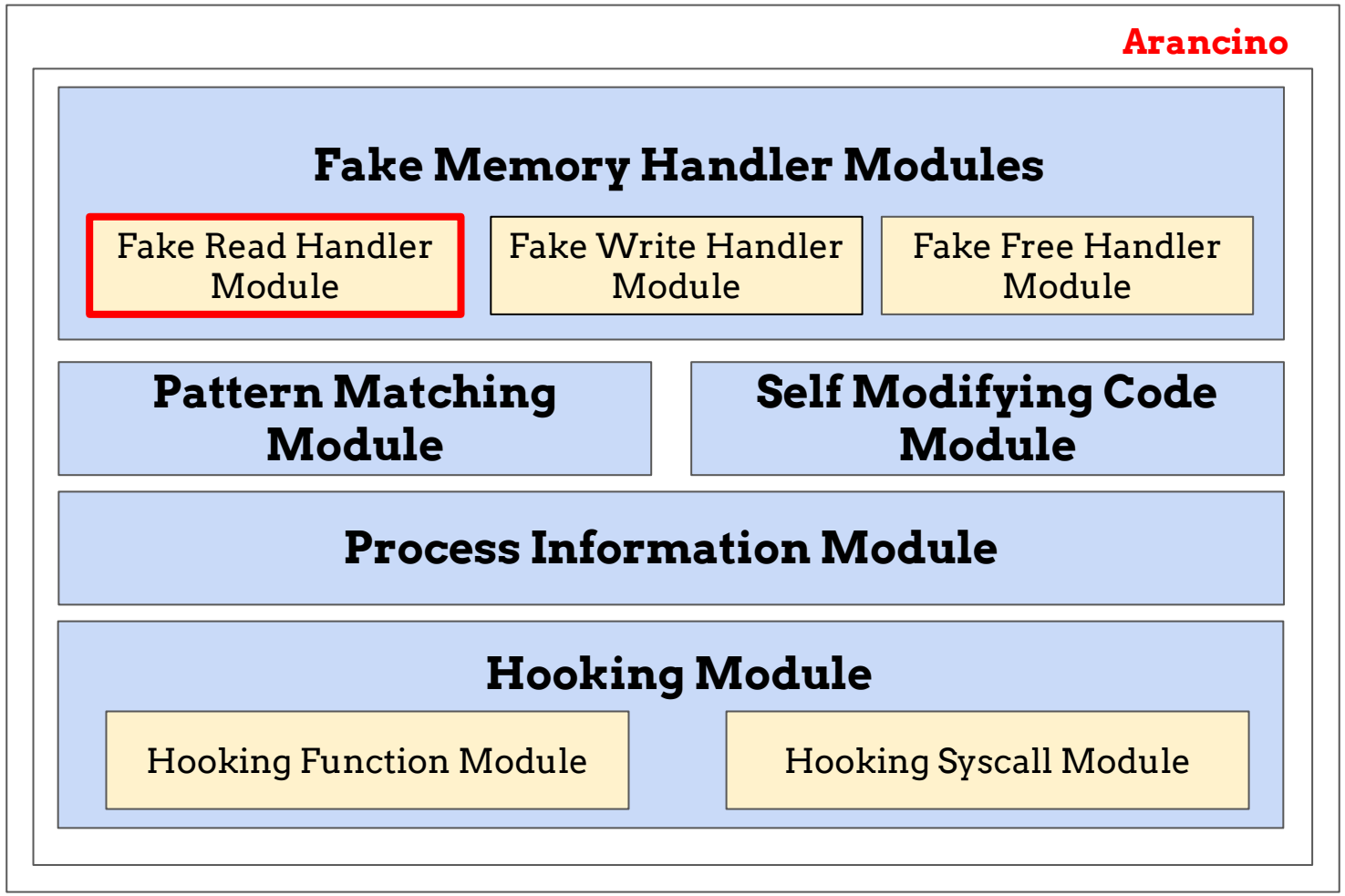
```
77C76F58   8D8424 DC020000   LEA EAX,DWORD PTR SS:[ESP+2DC]
77C76F5F   64:8B0D 00000000  MOV ECX,DWORD PTR FS:[0]
77C76F66   BA 406FC777       MOV EDX,ntdll.77C76F40
77C76F6B   8908              MOV DWORD PTR DS:[EAX],ECX
```

`KiUserApcDispatcher` - **normal execution**

```
77C76F58   E9 839CA0E3       JMP 5B680BE0
77C76F5D   0000              ADD BYTE PTR DS:[EAX],AL
77C76F5F   64:8B0D 00000000  MOV ECX,DWORD PTR FS:[0]
77C76F66   BA 406FC777       MOV EDX,ntdll.77C76F40
```

`KiUserApcDispatcher` - **Instrumented execution**

Arancino

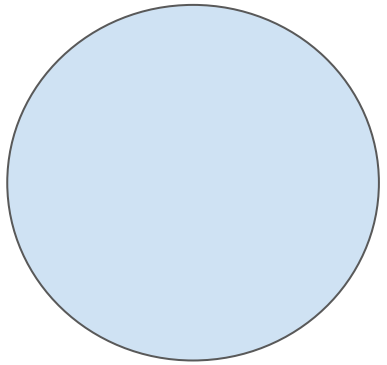

Arancino
Fake Memory Handler Modules
Fake Read Handler Module
Fake Write Handler Module
Fake Free Handler Module
Pattern Matching Module
Self Modifying Code Module
Process Information Module
Hooking Module
Hooking Function Module
Hooking Syscall Module

POLITECNICO MILANO 1863

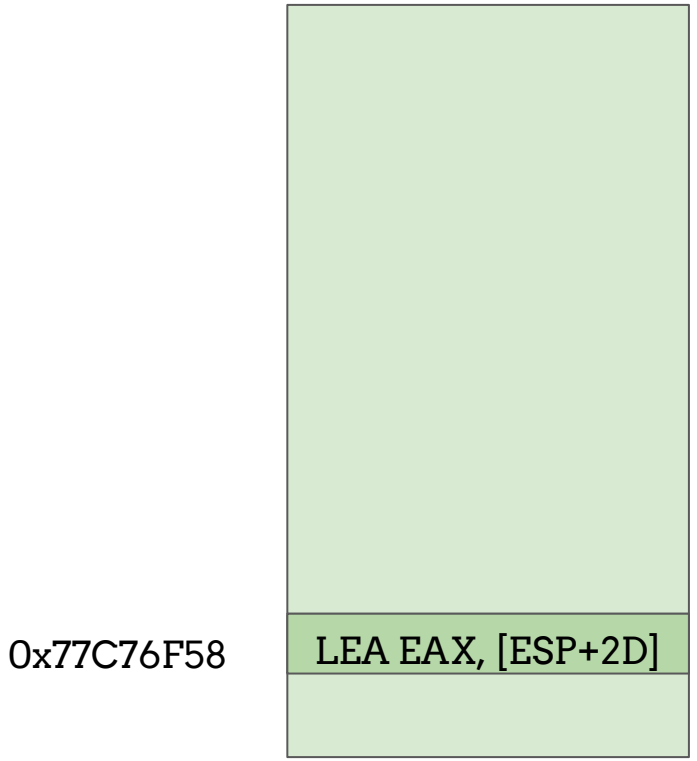
JITC Detection - DLL Hook

TRACE

FAKE_READ_HANDLER



MEMORY

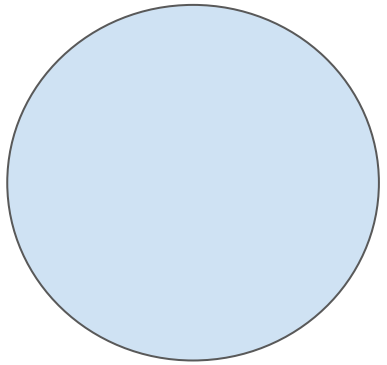


POLITECNICO
MILANO 1863

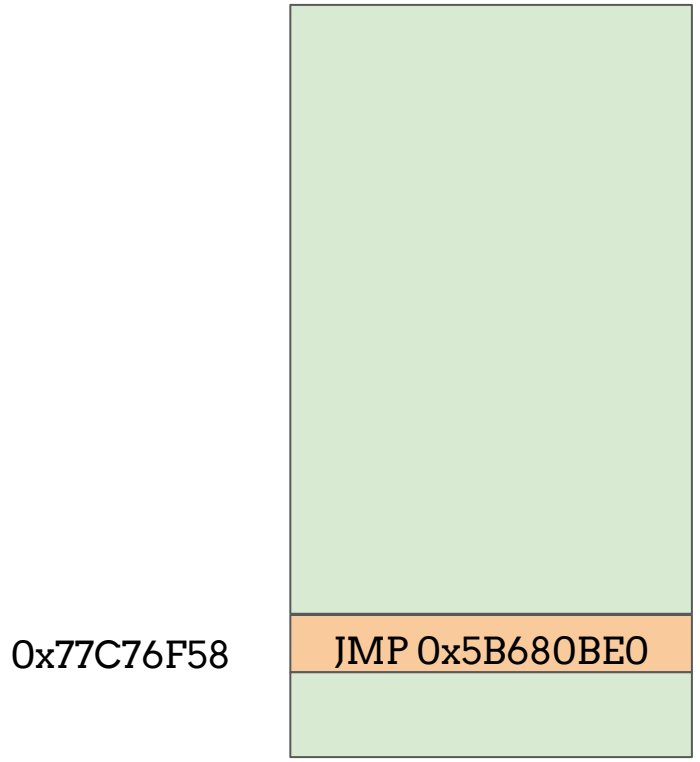
JITC Detection - DLL Hook

TRACE

FAKE_READ_HANDLER



MEMORY



POLITECNICO MILANO 1863

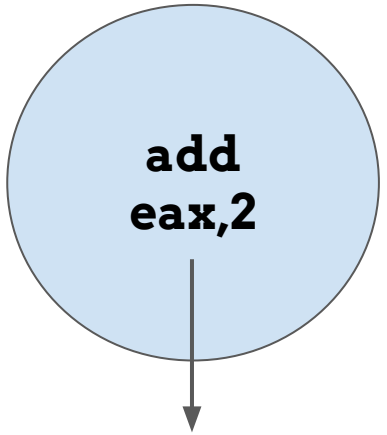
JITC Detection - DLL Hook

TRACE

add eax,2
mov edx, [eax]
cmp edx,0x8d
jnz ebx

eax = 0x77C76F58

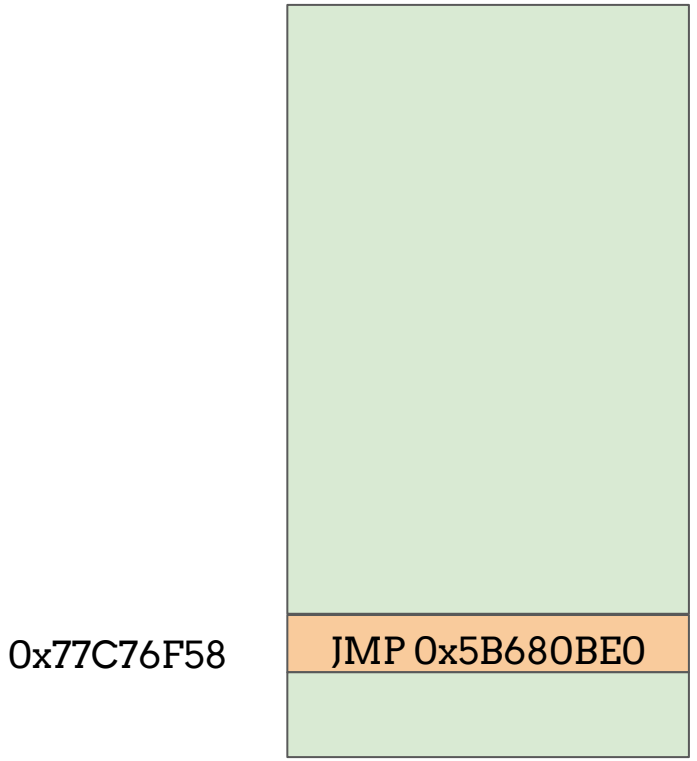
FAKE_READ_HANDLER



Is memory read operation?

Nope!
Go next

MEMORY



POLITECNICO MILANO 1863

JITC Detection - DLL Hook

TRACE

add eax,2
mov edx, [eax]
cmp edx,0x8d
jnz ebx

eax = 0x77C76F58

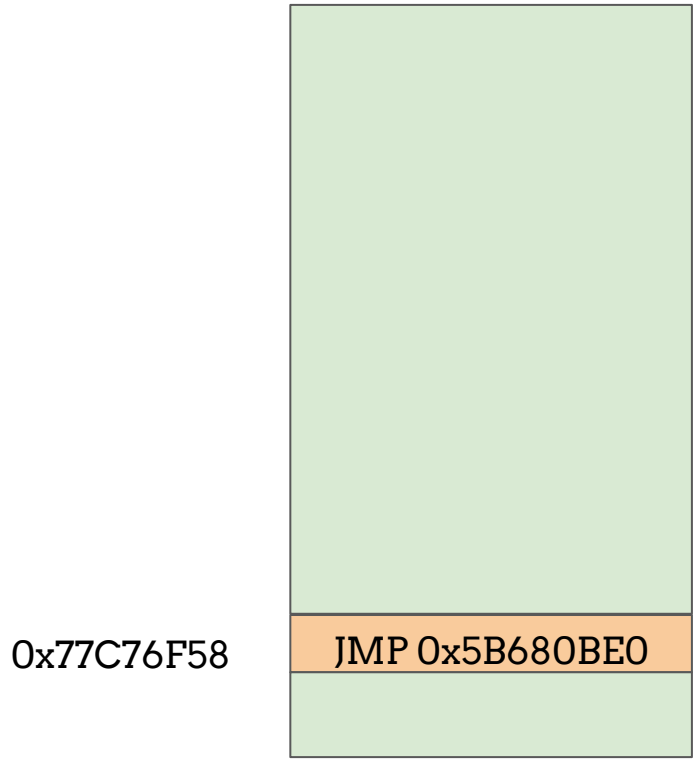
FAKE_READ_HANDLER

mov edx, [eax]

Is memory read operation?

✓ Yes

MEMORY



POLITECNICO MILANO 1863

JITC Detection - DLL Hook

TRACE

add eax,2
mov edx, [eax]
cmp edx,0x8d
jnz ebx

eax = 0x77C76F58

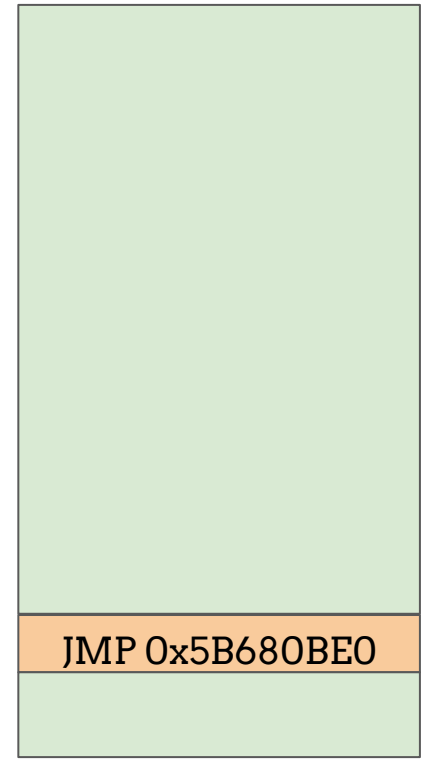
FAKE_READ_HANDLER

mov edx, [eax]

Is the target address inside a fake memory item?

Yes
fake memory function invoked

MEMORY



POLITECNICO MILANO 1863

JITC Detection - DLL Hook

TRACE

add eax,2
mov edx, [eax]
cmp edx,0x8d
jnz ebx

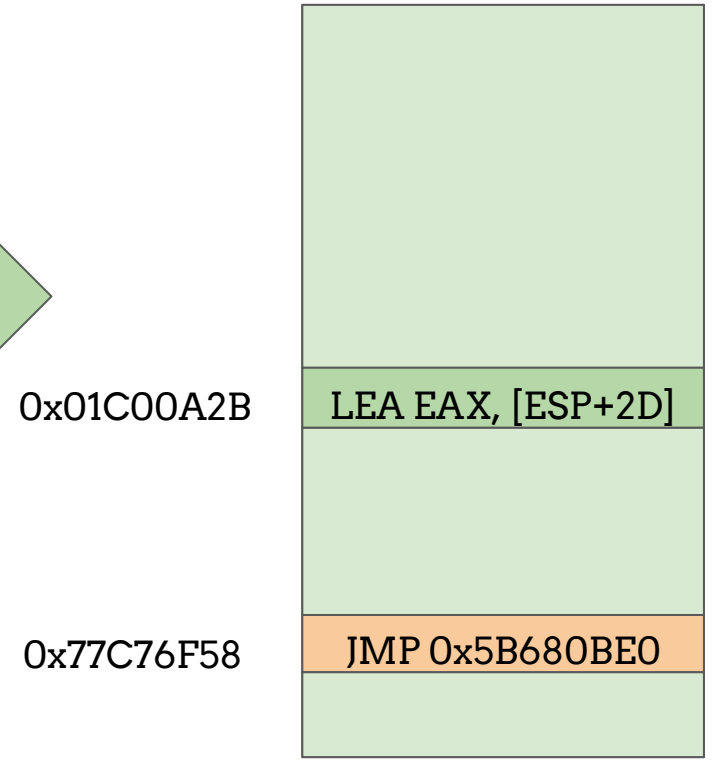
eax = 0x77C76F58

FAKE_READ_HANDLER

mov edx, [eax]

FakeMemoryFunc()
0x77C76F58
0x77C76F5F

MEMORY



POLITECNICO MILANO 1863

JITC Detection - DLL Hook

TRACE

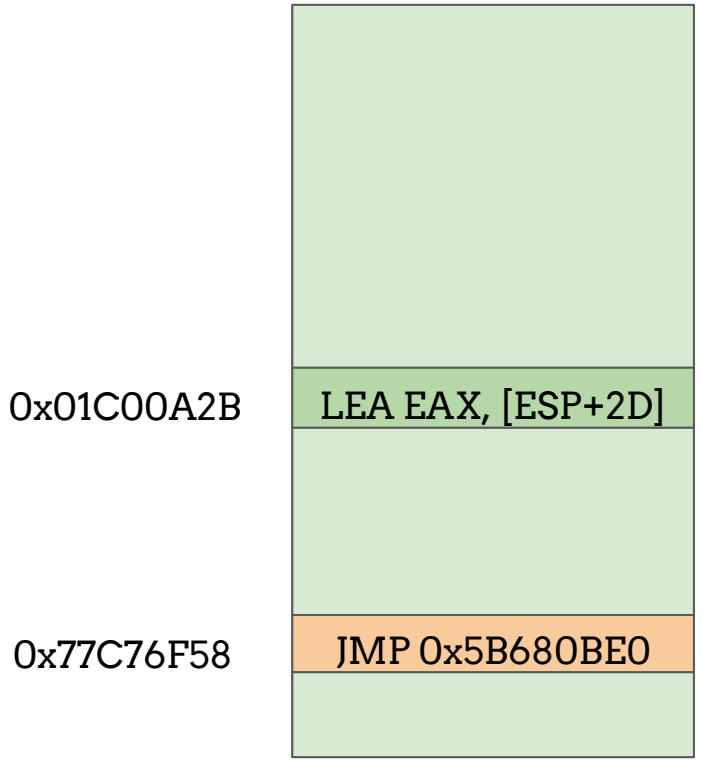
add eax,2
mov edx, [eax]
cmp edx,0x8d
jnz ebx

Instrumented process read the fake value:
LEA EAX, [ESP+2D]
and doesn't detect PIN

FAKE_READ_HANDLER

mov edx, [eax]

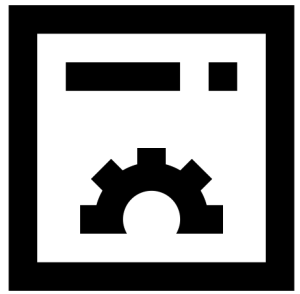
MEMORY



POLITECNICO
MILANO 1863

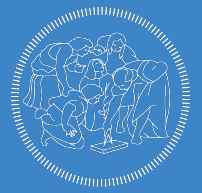
JIT Compiler Detection

- **Memory Page Permissions**
  - Checks if there are **WX pages**



- **DLL Hook Detection**

- **Memory Allocations**



POLITECNICO
MILANO 1863

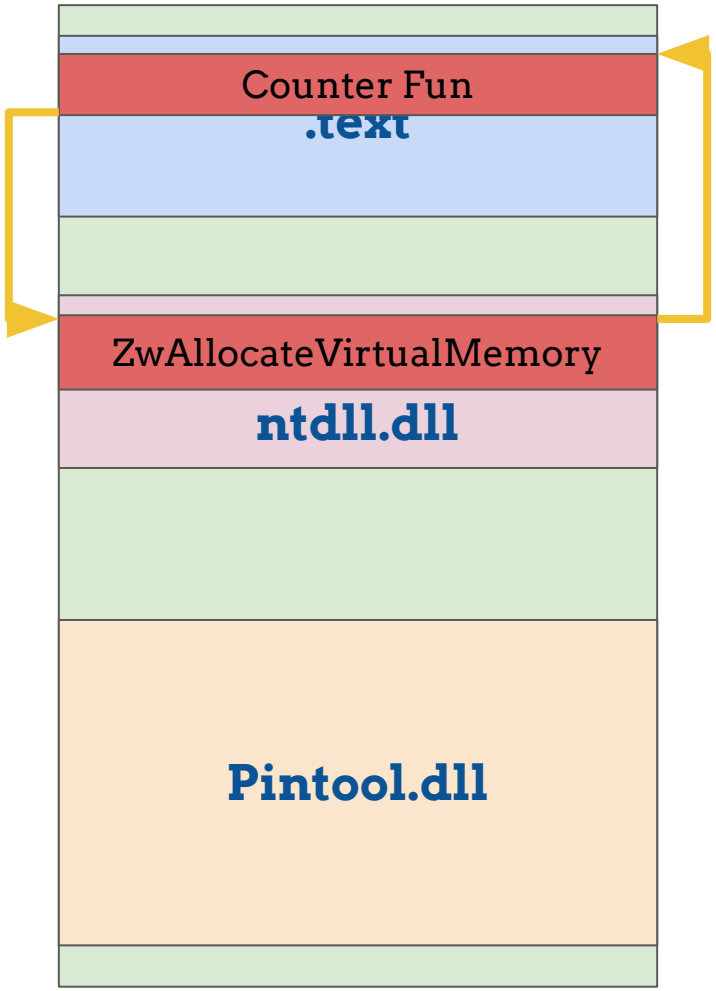
JIT Compiler - API Hook

JIT Compiler needs **Memory** to perform the compiling

We can monitor the allocation by Hooking at
ZwAllocateVirtualMemory

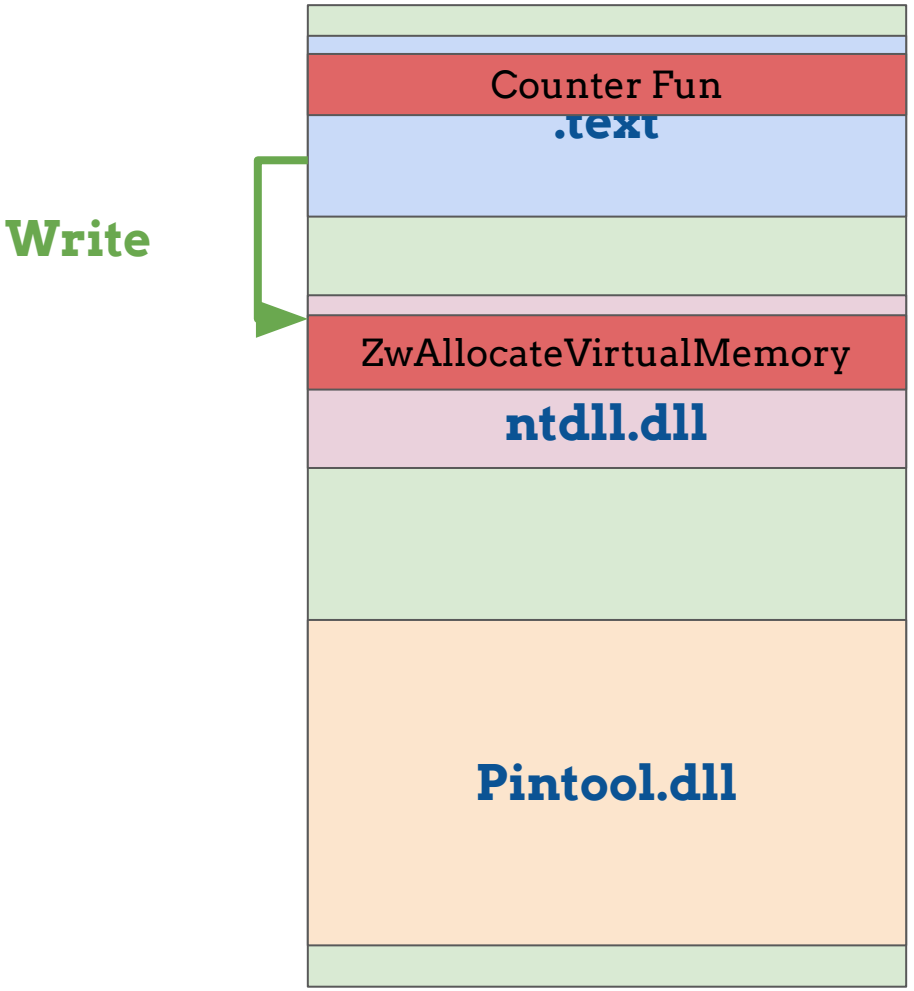
POLITECNICO
MILANO 1863

JIT Compiler - API Hook

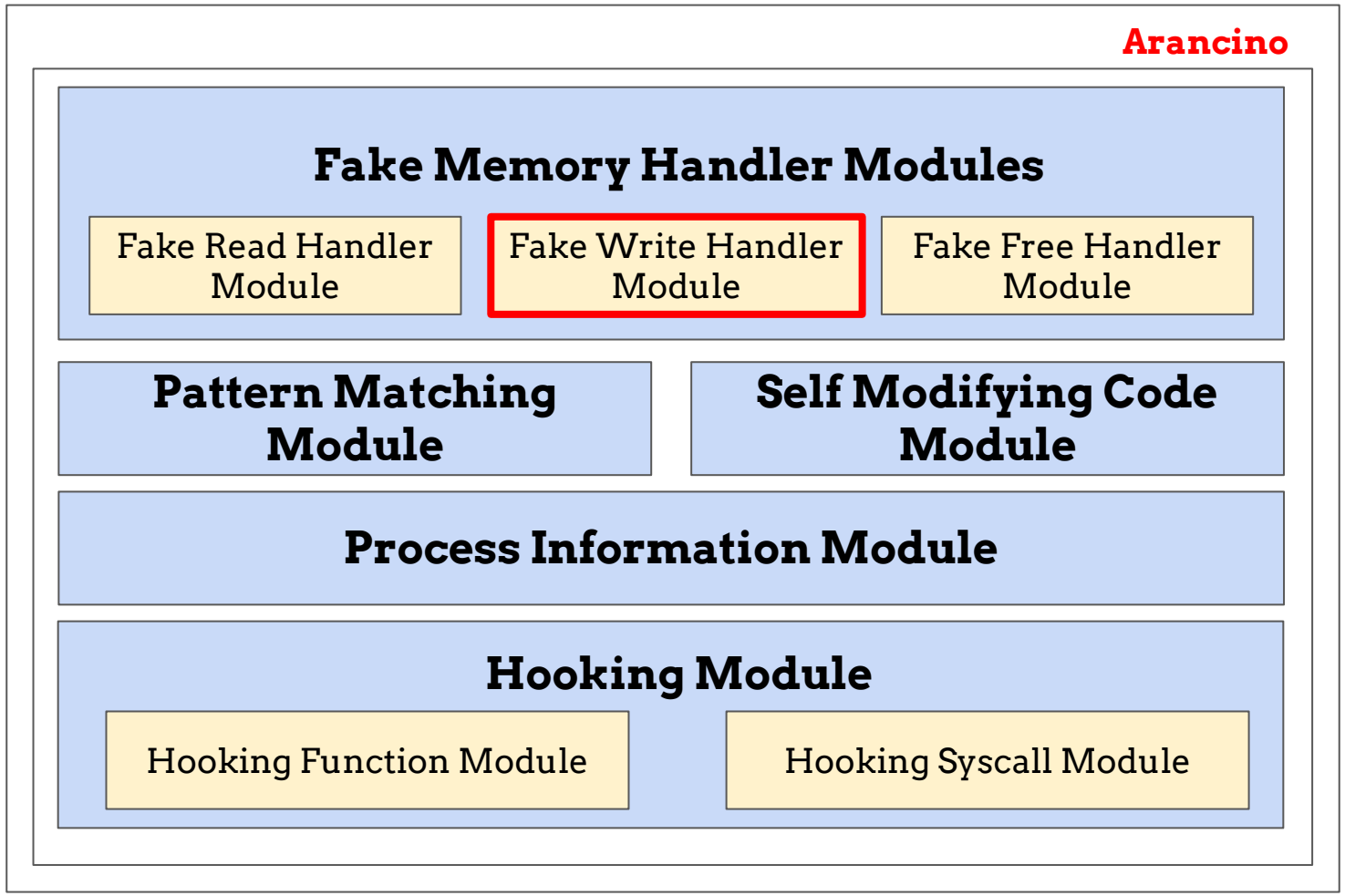


POLITECNICO
MILANO 1863

JIT Compiler - API Hook

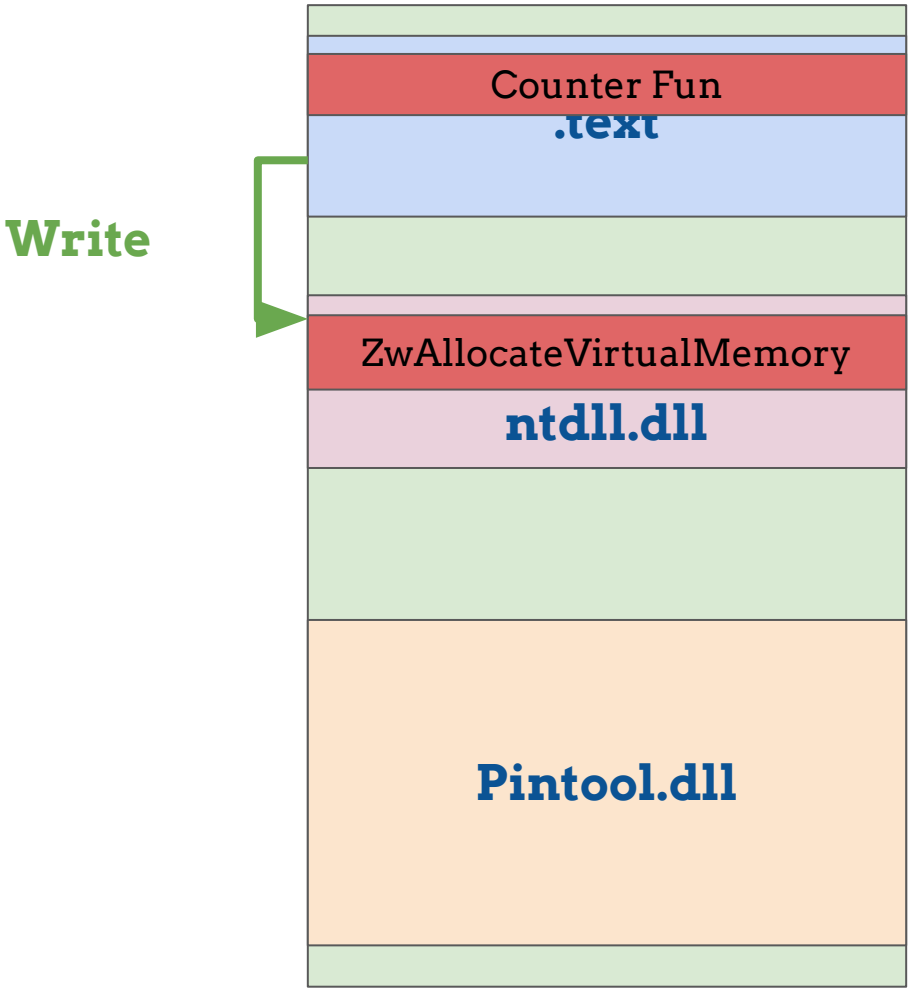


Arancino


Arancino
Fake Memory Handler Modules
Fake Read Handler Module
Fake Write Handler Module
Fake Free Handler Module
Pattern Matching Module
Self Modifying Code Module
Process Information Module
Hooking Module
Hooking Function Module
Hooking Syscall Module

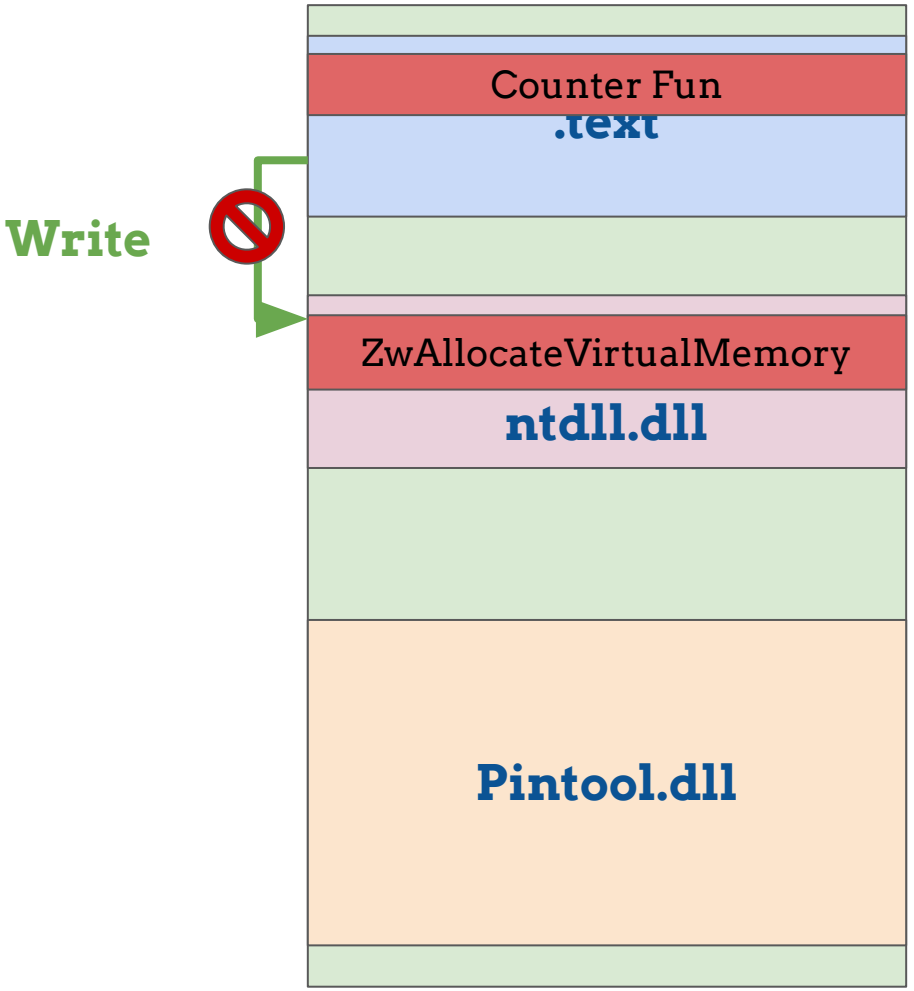
POLITECNICO
MILANO 1863

JIT Compiler - API Hook



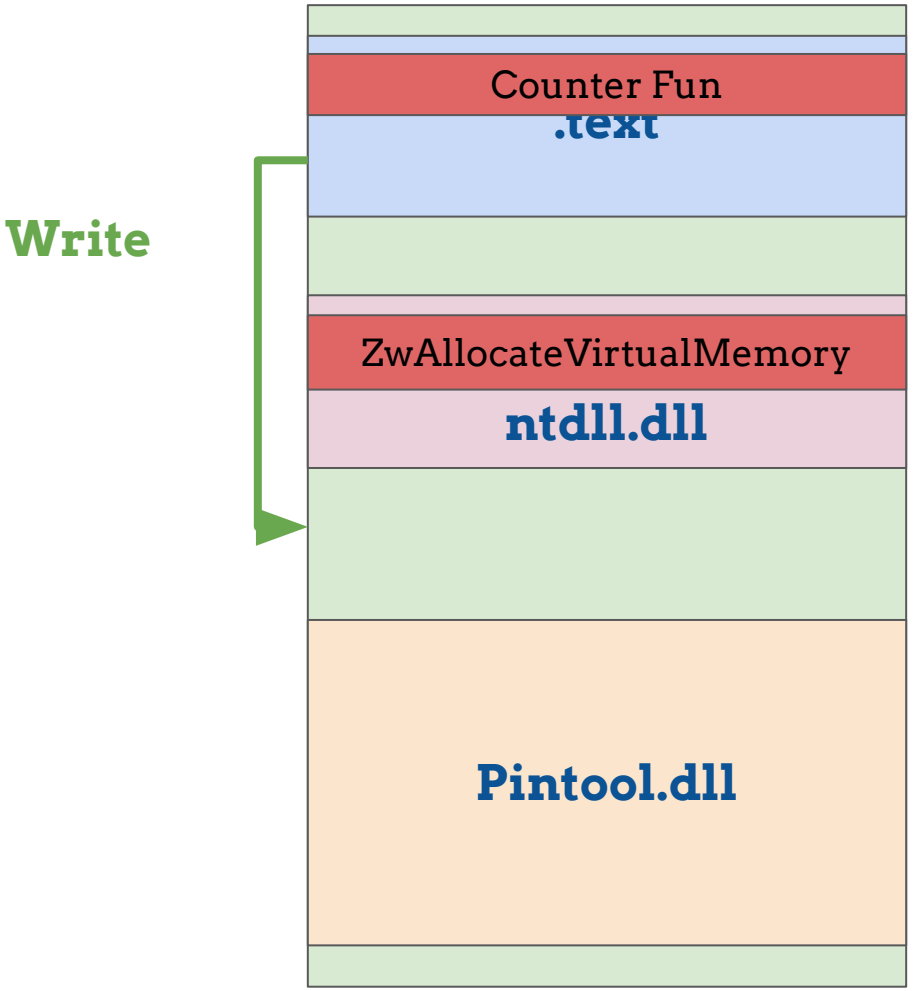
POLITECNICO
MILANO 1863

JIT Compiler - API Hook



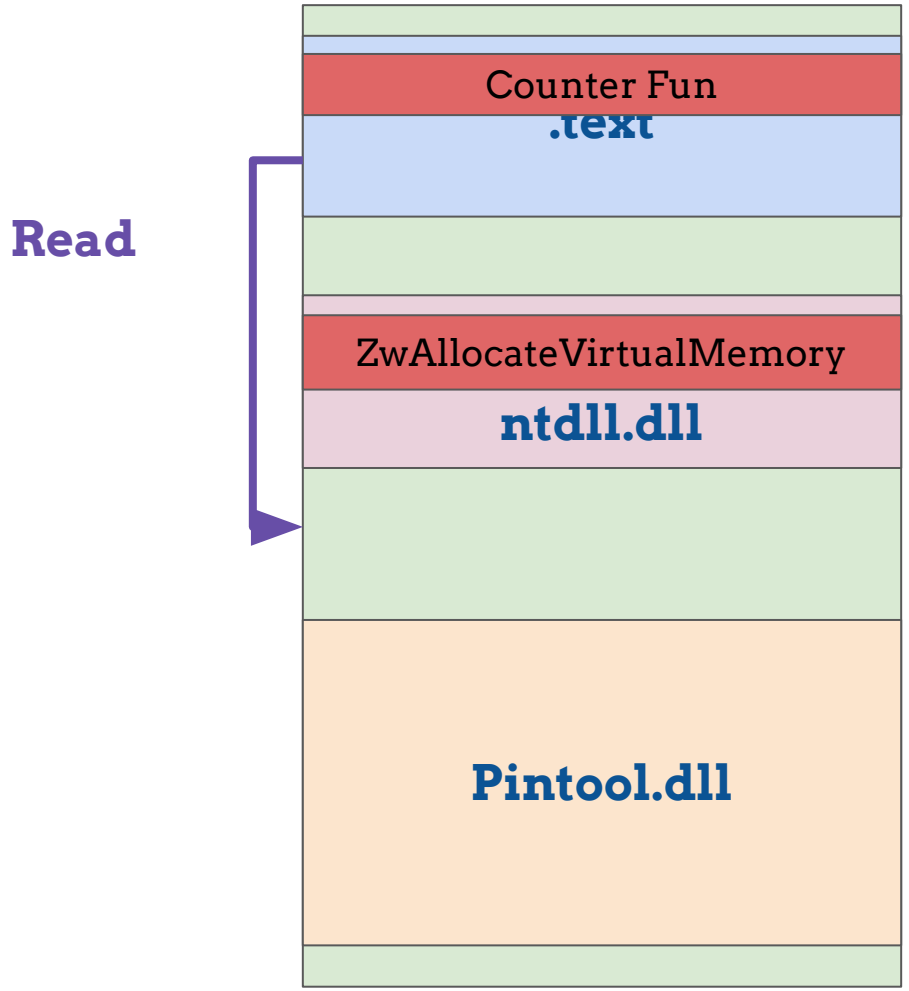
POLITECNICO
MILANO 1863

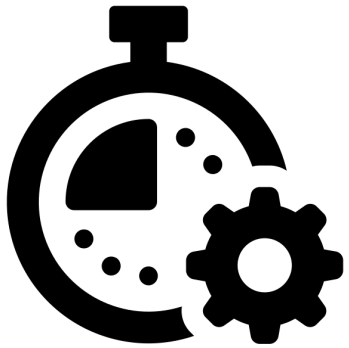
JIT Compiler - API Hook



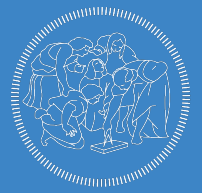
POLITECNICO
MILANO 1863

JIT Compiler - API Hook





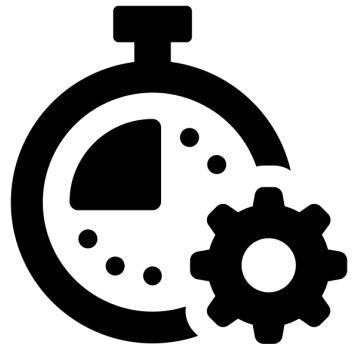
Overhead Detection



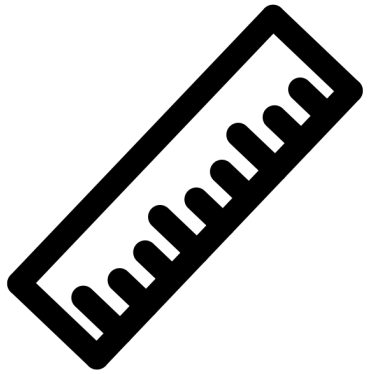
POLITECNICO
MILANO 1863

Overhead Detection

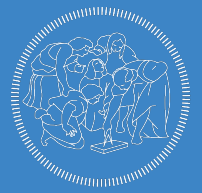
- **Windows Time**
  - Use windows API
    - GetTickCount and timeGetTime
  - Or Windows Structures
    - KUSER_SHARED_DATA.



- **CPU Time**
  - Count CPU cycles (`rdtsc`)



Evasive Malware Measurement

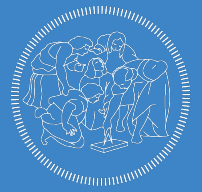


POLITECNICO
MILANO 1863

Anti-Instrumentation Measurement

Dataset

- **7006** Binaries
- Virus Total Intelligence (3+ AV Detection)
- From October 2016 to February 2017

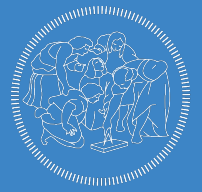


POLITECNICO
MILANO 1863

Anti-Instrumentation Measurement

Environment Setup

- **Virtual Machine** (VirtualBox)
- **Windows 7** (64-bit)
- **Custom Apps** (Adobe Reader, Chrome, and media players)
- **User Data** (saved credentials, browser history, etc.)
- **Basic User Activity** (moving the mouse, launching applications)
- **5 min** run



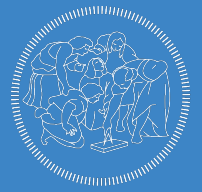
POLITECNICO
MILANO 1863

Evasive Malware

At least one evasive behavior: 1,093 / 7006 (15.6%)

Family Name [1]	Samples	Evasive	Techniques
virlock	619 (8.8%)	600 (96.9%)	2
confidence	505 (7.2%)	68 (13.5%)	4
virut	242 (3.5%)	13 (5.4%)	2
mira	230 (3.3%)	9 (3.9%)	1
upatre	187 (2.7%)	2 (1.1%)	1
lamer	171 (2.4%)	0 (0.0%)	0
sivis	168 (2.4%)	0 (0.0%)	0

[1] **AvClass** https://github.com/malicialab/avclass



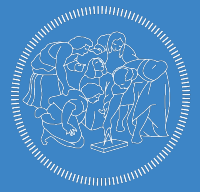
POLITECNICO
MILANO 1863

Top Evasive Malware

At least one evasive behavior: 1,093 / 7006 (15.6%)

Family Name [1]	Samples	Evasive	Techniques
sfone	19	19 (100.0%)	1
unruy	11	11 (100.0%)	1
virlock	619	600 (96.9%)	2
vilsel	13	8 (61.5%)	2
urelas	18	9 (47.4%)	2
confuser	52	8 (44.4%)	1
vobfus	29	19 (36.5%)	1

[1] **AvClass** https://github.com/malicialab/avclass

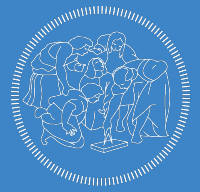


POLITECNICO
MILANO 1863

Top Techniques Used

At least one evasive behavior: 1,093 / 7006 (15.6%)

	Technique	#
Code Cache Artifacts	Self-modifying code	897
Environment Artifacts	Parent detection	259
JIT Compiler Detection	Write on protected memory region	40
Environment Artifacts	Check DEBUG flag	5
Environment Artifacts	Memory fingerprinting	3



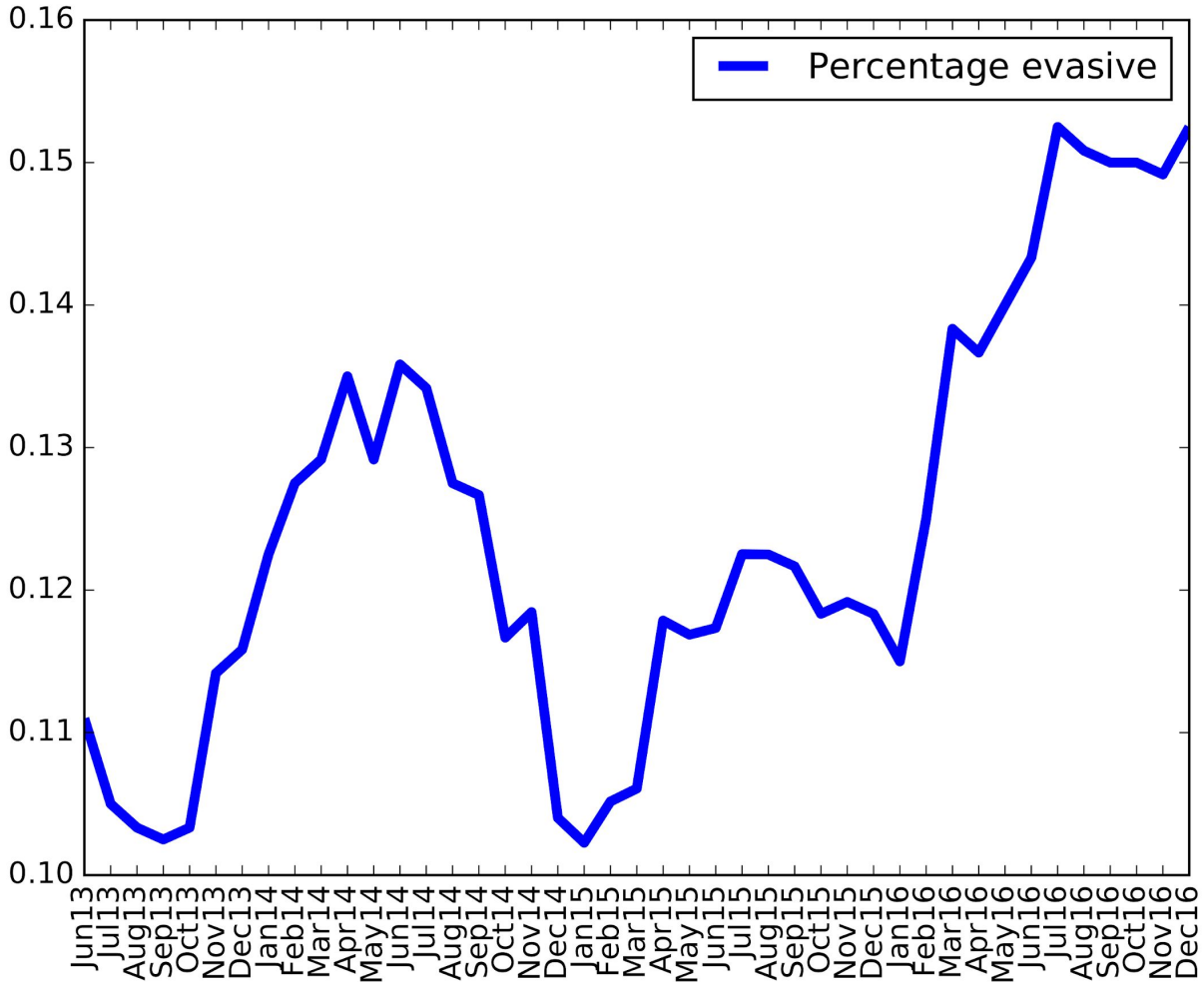
POLITECNICO
MILANO 1863

Overhead

	Pin time [ms]	Arancino [ms]	Arancino overhead [%]	Module activated
Parent Detection	850	870	2%	Hooking Module
EIP Detection - int2e	710	1,150	62%	Pattern Match Module
Memory Fingerprinting	2,000	7,090	254,5%	Fake Read Module
Memory Allocations	2,000	2,900	45%	Fake Write Module + Hooking Module

POLITECNICO
MILANO 1863

Evasive Malware Timeline



Thanks!

https://github.com/necst/arancino

Mario Polino

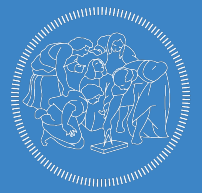
<mario.polino@polimi.it>

Questions?

https://github.com/necst/arancino

Mario Polino

<mario.polino@polimi.it>



POLITECNICO
MILANO 1863

Credits

- Icons, CC from Noun Project:
  - Vicons Design
  - Aya Sofya
  - Adnen Kadri
  - Stock Image Folio
  - Icon Fair
  - Creative Stall
  - Gregor Cresnar