SECURITY FEATURES

1. CONTROL FLOW INTEGRITY
   A. ROP ATTACK MITIGATION
   B. COP/ JOP ATTACK MITIGATION
2. POINTER SAFETY
3. ENCRYPTED ISLANDS

GOALS:

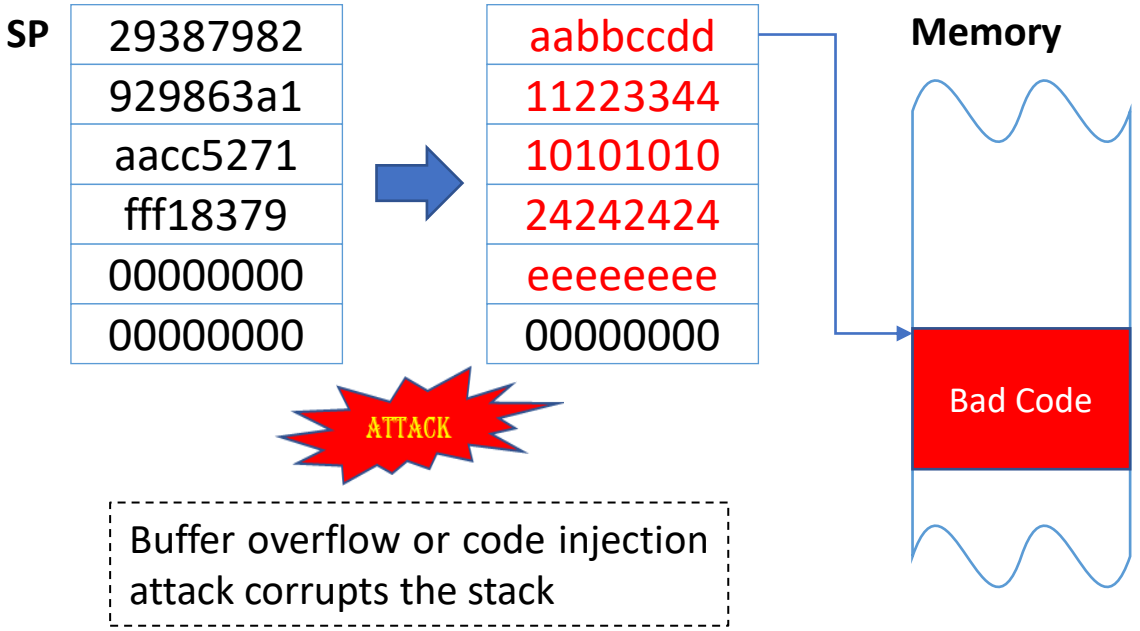
- **ELEVATE RISC-V SECURITY BY INTRODUCING ARCHITECTURAL FEATURES VIA SIMPLE ISA EXTENSIONS**
- **ENSURE THERE IS NO PATENT/ IP INFRINGEMENT IN ANY WAY/ CO-DEVELOP WITH COMMUNITY PARTNERS**

1A. CFI – ROP ATTACK MITIGATION

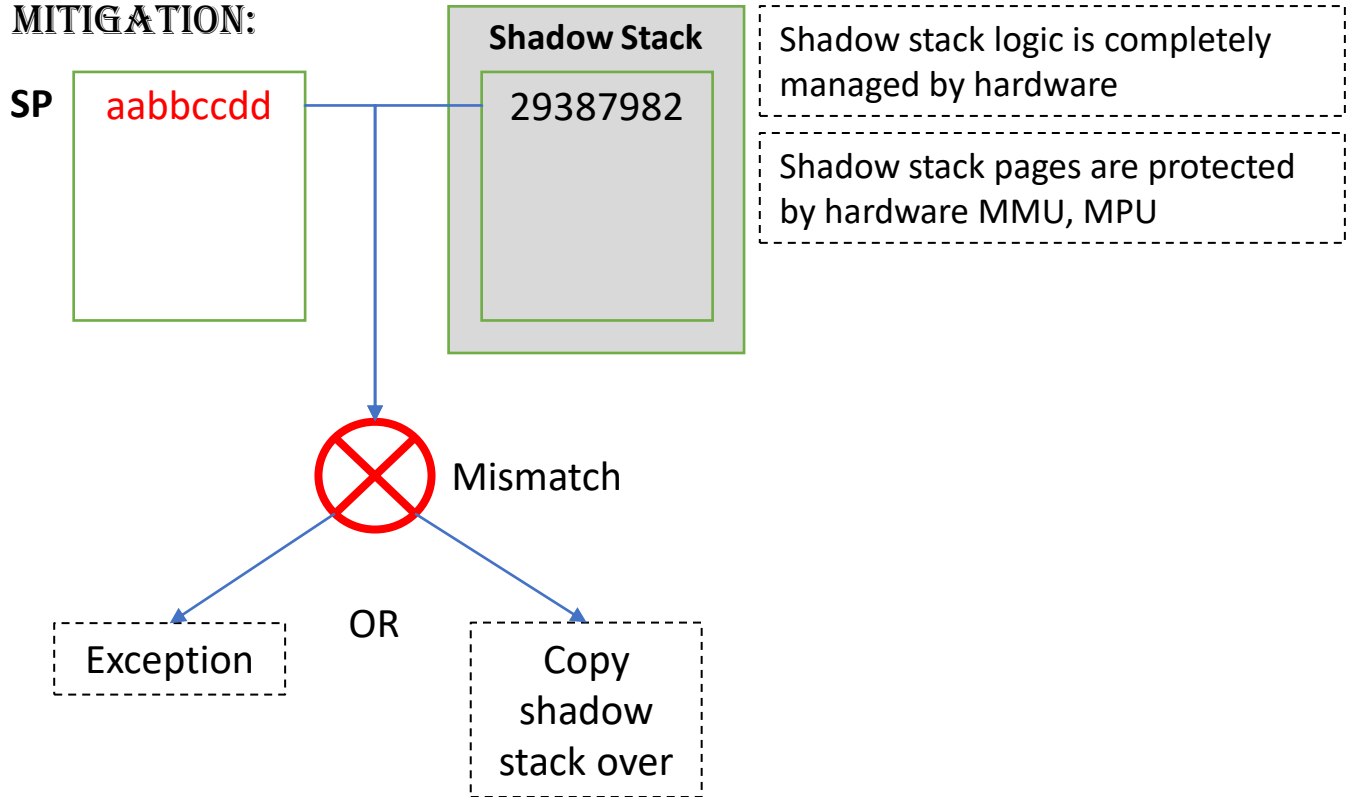
THREAT MODEL:

Asset	Description	Security Property C - Confidentiality I - Integrity A - Availability	Threat	Entry Point of Threat	Impact of Vulnerability	Severity (CVSS v3 Rating) https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator	Mitigation/ Security Requirement
Stack	System Stack	I	Tamper	Stack smashing by either buffer overrun or injecting code into the stack – Return Oriented Programming (ROP) attacks	Control flow hijack	HIGH: 7.5 **CVSS v3.1 Vector** AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N	Use shadow stack to compare return addresses for control flow transfer instructions, if mismatch then raise exception or copy shadow stack over

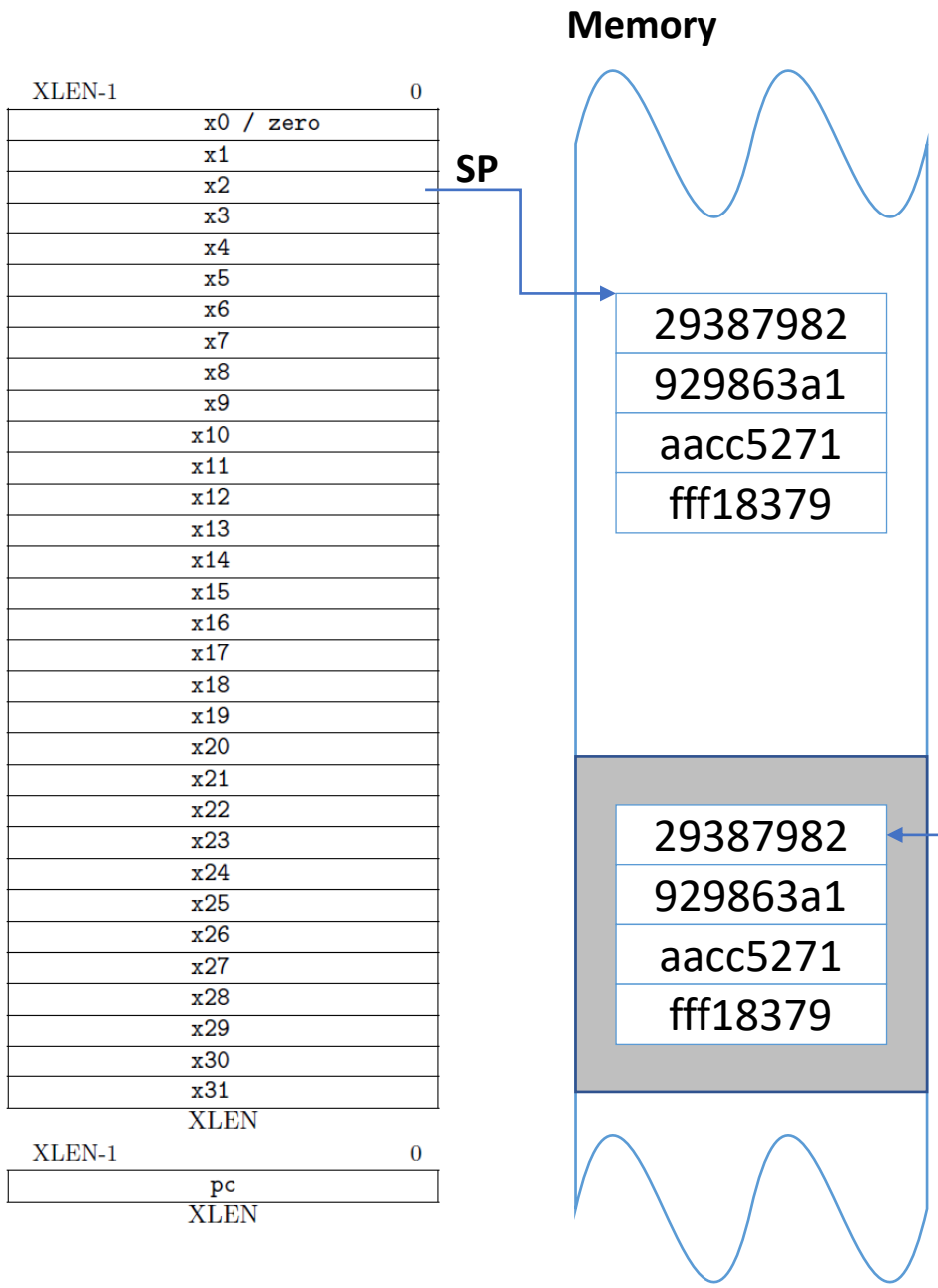
ATTACK SCENARIO:



MITIGATION:



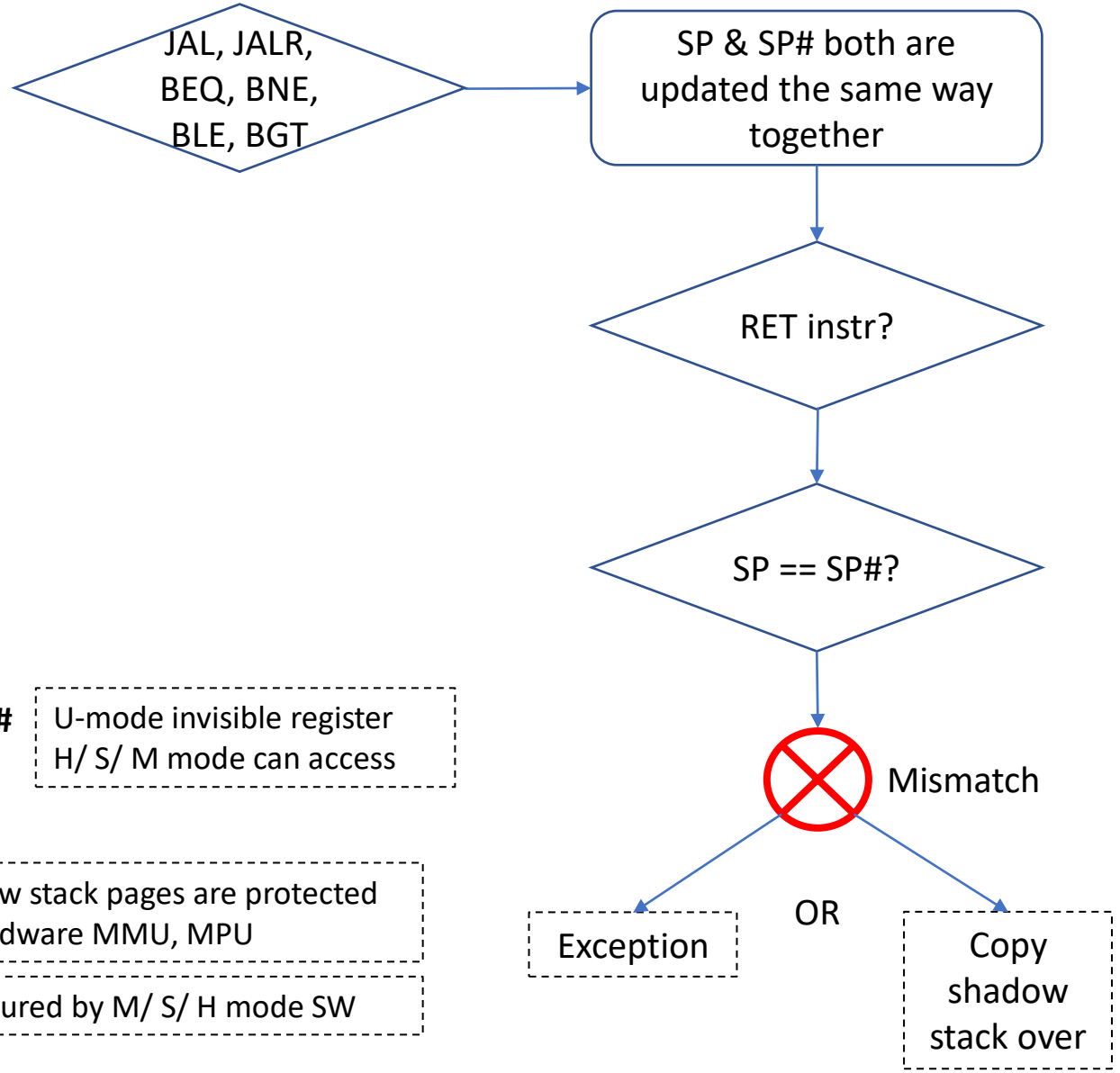
1A. CFI – ROP ATTACK MITIGATION



U-mode invisible register
H/ S/ M mode can access

Shadow stack pages are protected by hardware MMU, MPU

Configured by M/ S/ H mode SW



1. SP# shall also be considered part of a process/ VM's context space
2. M/ S/ H mode software needs to update SP# as part mode/ context switch

Figure 2.1: RISC-V base unprivileged integer register state.

1A. CFI – ROP ATTACK MITIGATION

PROPOSED PRIVILEGE ISA EXTN

Mnemonic: SLOAD
Opcode: TBD

Operation: Only S/ H/ M mode software can use this instruction to load stack address to the shadow stack pointer at every mode/ context switch

Behavior: S/ H/ M mode – valid; U mode - trapped

Note: Loading this shadow stack pointer is not enough, but also the shadow stack needs to be protected with MMU/ MPU via configuration

1B. CFI – COP/ JOP ATTACK MITIGATION

THREAT MODEL:

Asset	Description	Security Property C - Confidentiality I - Integrity A - Availability	Threat	Entry Point of Threat	Impact of Vulnerability	Severity (CVSS v3 Rating) https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator	Mitigation/ Security Requirement
Call/ Jump Targets	Indirect call/ jump target addresses	I	Tamper	Tampering the code to perform indirect call/ jump to invalid locations	Control flow hijack	HIGH: 7.5 **CVSS v3.1 Vector** AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N	Track indirect call/ jump instructions and permitting only valid call/ jump locations of the code with special instruction and state machine

ATTACK SCENARIO:

```
main() {
   int (*pFunc)();
   pFunc = test;
   pFunc();
}

int test() {
   if(cond1 == X)
      return 1;
   else if (cond2 == Y)
      return 2;
   return 0;
}
```

```
<main>:
j test
jr ra
:
<test>:
bne x12, x11, else
j done
bne x13, x12, done
done:
jr ra
```

```
<main>:
j test+4
jr ra
:
<test>:
bne x12, x11, else
j done
bne x13, x12, done
done:
jr ra
```

ATTACK

Bypassed

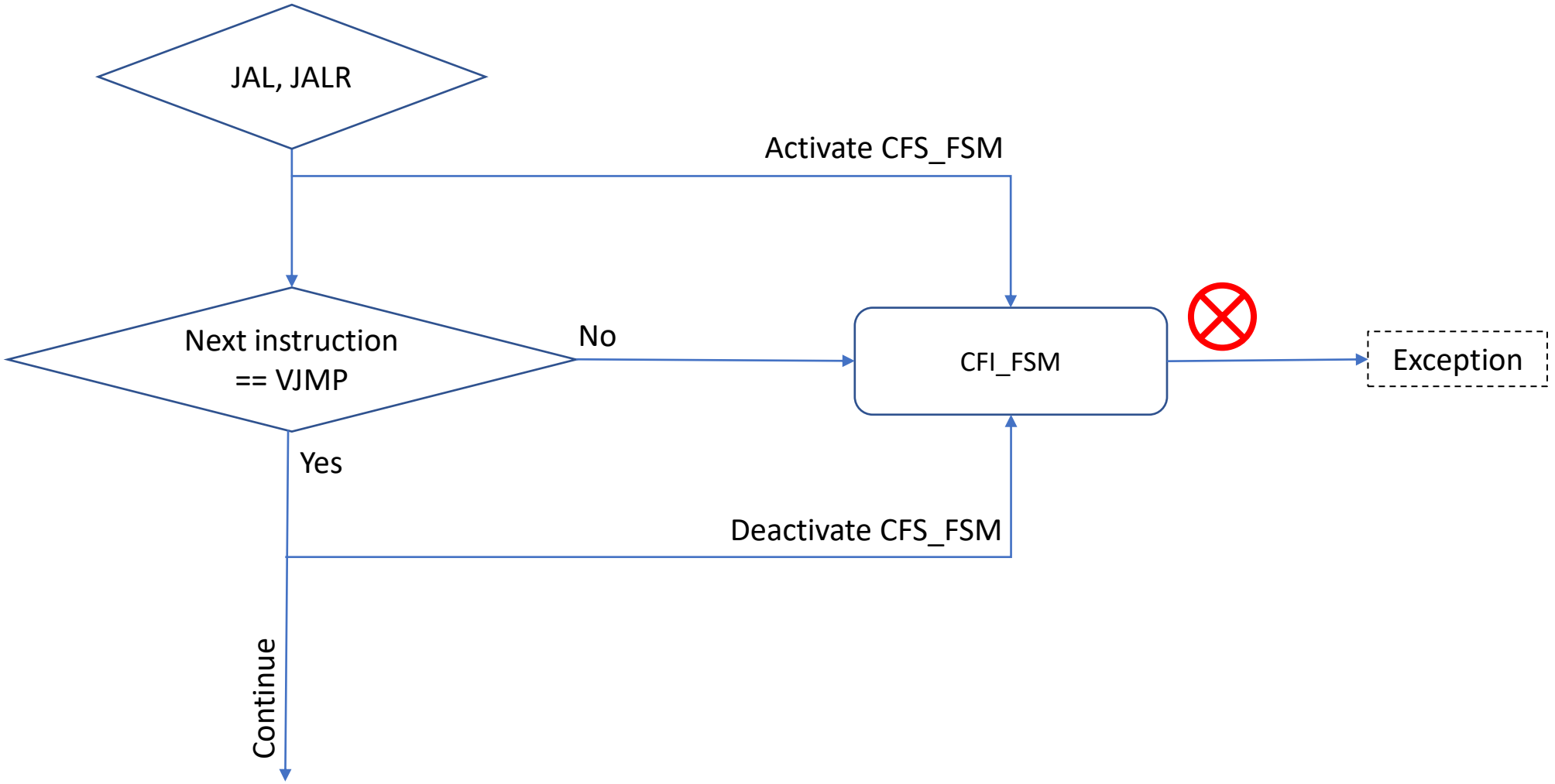
MITIGATION:

```
<main>:
VJMP
j test
jr ra
:
<test>:
VJMP
bne x12, x11, else
j done
bne x13, x12, done
done:
jr ra
```

Compiler inserts a new marker instruction to indicate the address is a valid jump target address

Hardware checks the next subsequent instruction after an indirect jump or call, is a marker instruction or not, else it will fire an exception to report control flow integrity violation

1B. CFI – COP/ JOP ATTACK MITIGATION



1B. CFI – COP/ JOP ATTACK MITIGATION

PROPOSED PRIVILEGE ISA EXTN

Mnemonic: VJMP
Opcode: TBD

Operation: Its like a NOP instruction but CPU activates a state machine to look for if the next instruction is a VJMP or not. If yes, then continues execution. Else the execution is halted with an exception raised for control flow integrity violation

Behavior: S/ H/ M mode – valid; U mode - trapped

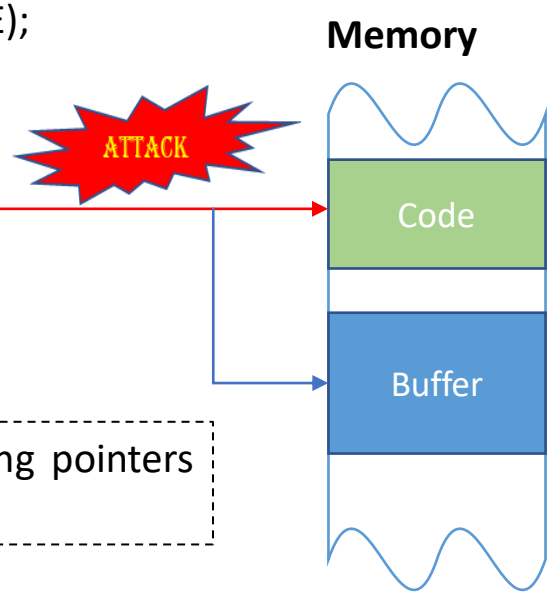
2. POINTER SAFETY

THREAT MODEL:

Asset	Description	Security Property C - Confidentiality I - Integrity A - Availability	Threat	Entry Point of Threat	Impact of Vulnerability	Severity (CVSS v3 Rating) https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator	Mitigation/ Security Requirement
Pointer	Memory Pointers	I	Tamper	Misusing the pointers to access illegal memory, manipulating stack, heap regions, executing data pointers, use after freeing, out of range access, etc.	Memory Safety	HIGH: 7.5 **CVSS v3.1 Vector** AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:N	Unused upper bits of pointer virtual address to hold, type, permissions and tag inserted by malloc function and checked by MMU during page walk to prevent memory misuse

ATTACK SCENARIO:

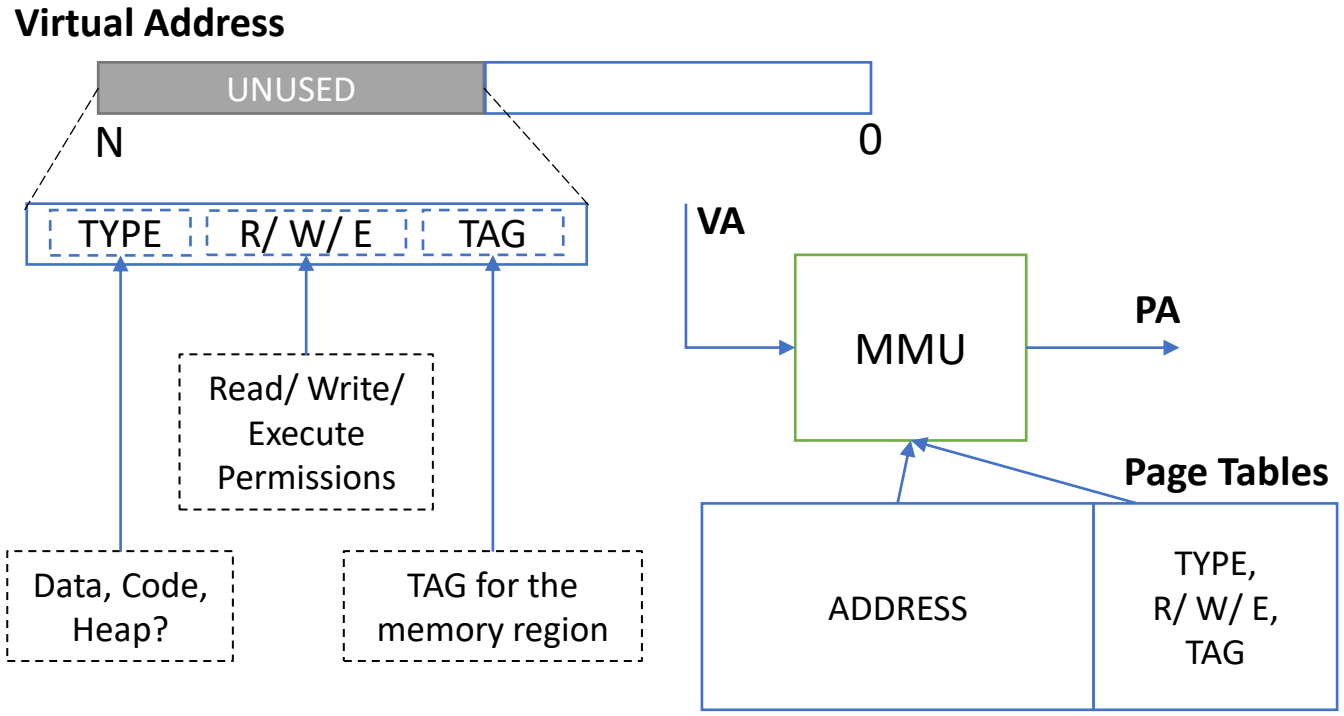
```
pBuf = malloc(16 * WORD_SIZE);
…
pBuf[500] = 0x1010;
…
```



Accessing illegal memory using pointers to tamper memory

MITIGATION:

SV32, SV39, SP48 virtual address formats have unused bits



2. POINTER SAFETY

POINTER CREATION

pBuf = malloc(16 * WORD_SIZE);

Memory Manager (Kernel)

Virtual Address

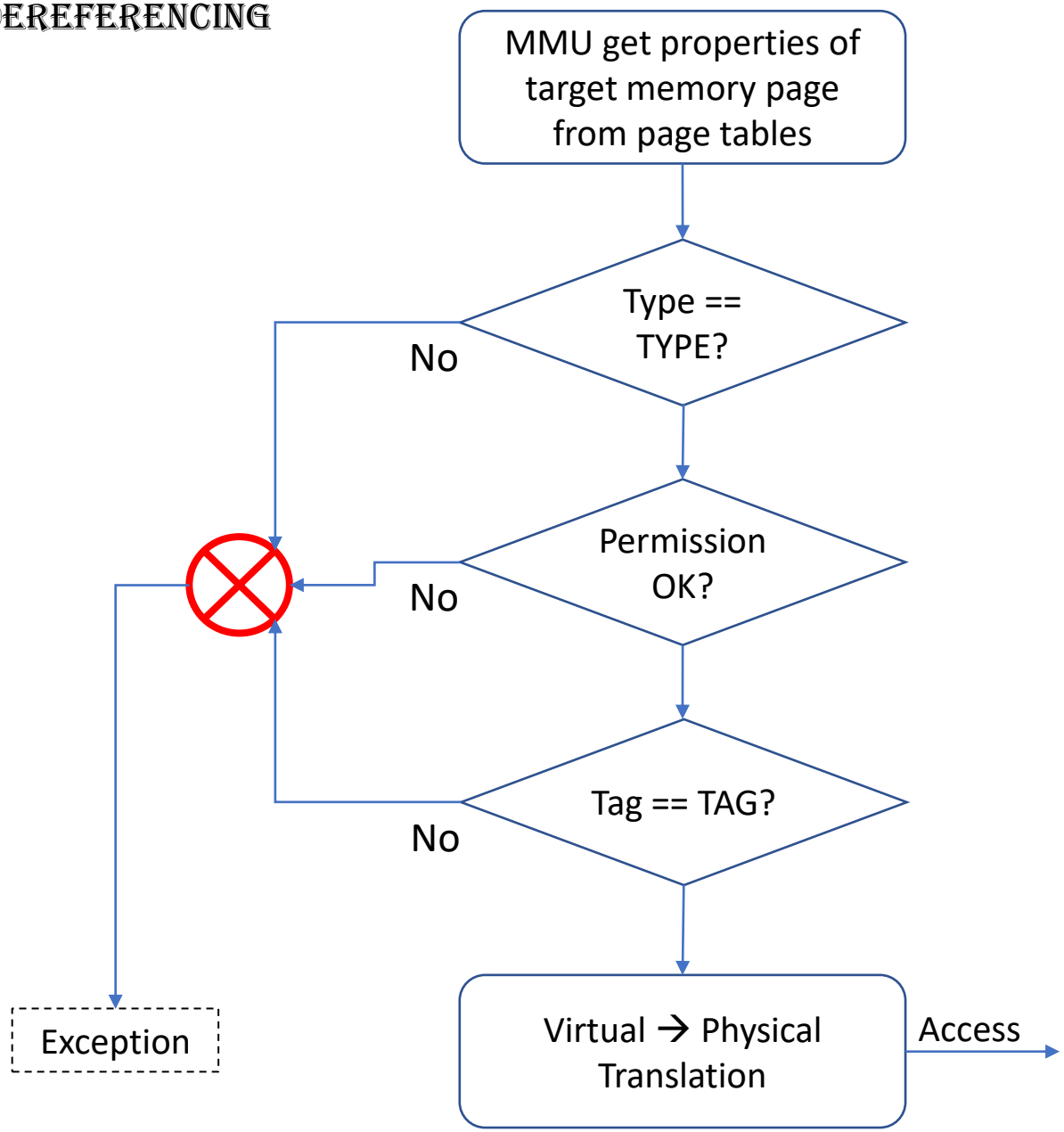
TYPE	R/ W/ E	TAG	0xfff18379
DATA	W	8	0xfff18379

Memory Manager (Kernel)

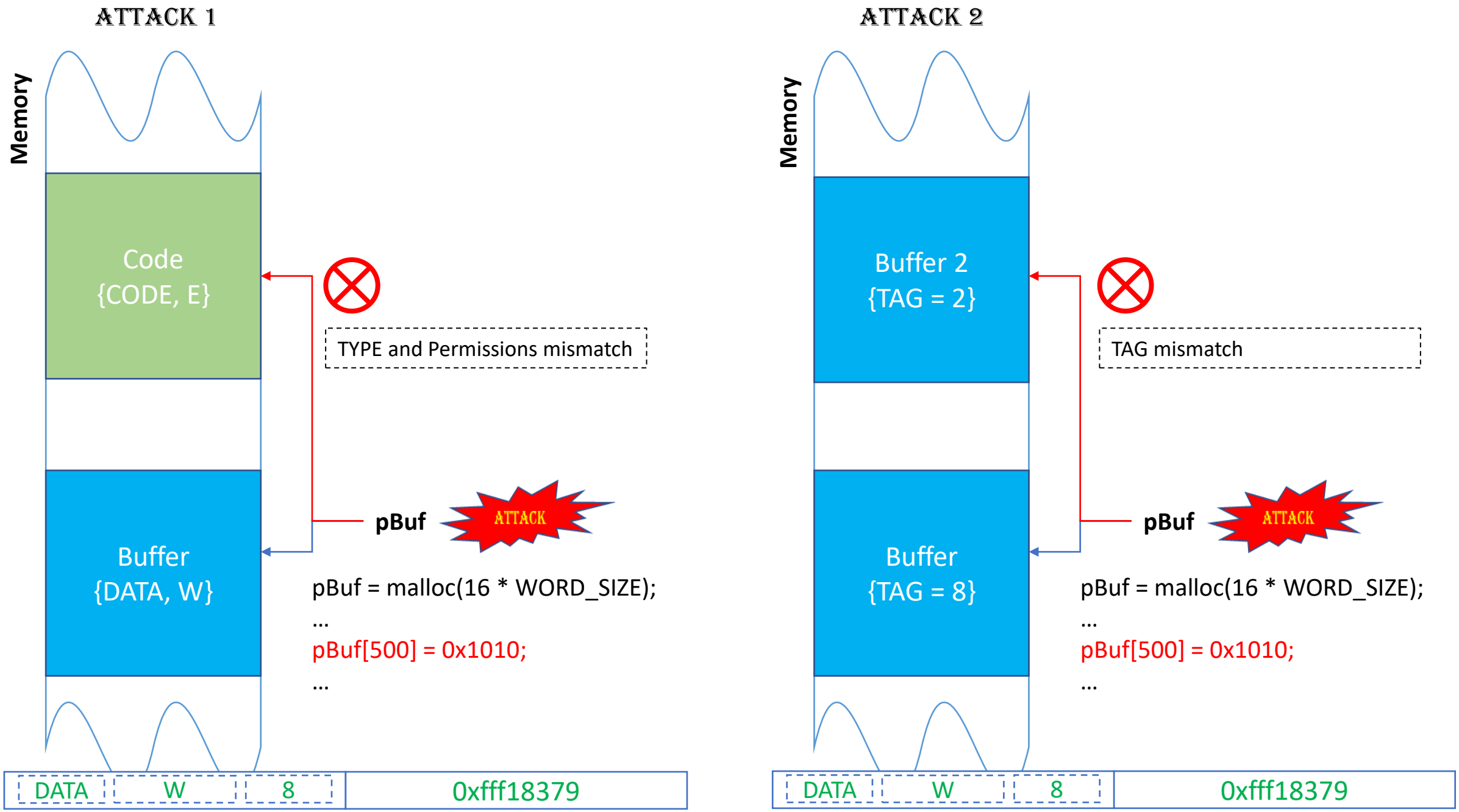
Page Tables

DATA	W	8	0xfff18379
HEAP	W	7	0xfff18112
DATA	W	6	0xfff1237e
DATA	W	e	0xfee18366
DATA	W	1	0xf0012345
CODE	E	4	0xff287482
DATA	W	3	0xf0028648
CODE	E	9	0xfff44778

DEREFERENCING



2. POINTER SAFETY



2. POINTER SAFETY

MMU EXTENSIONS

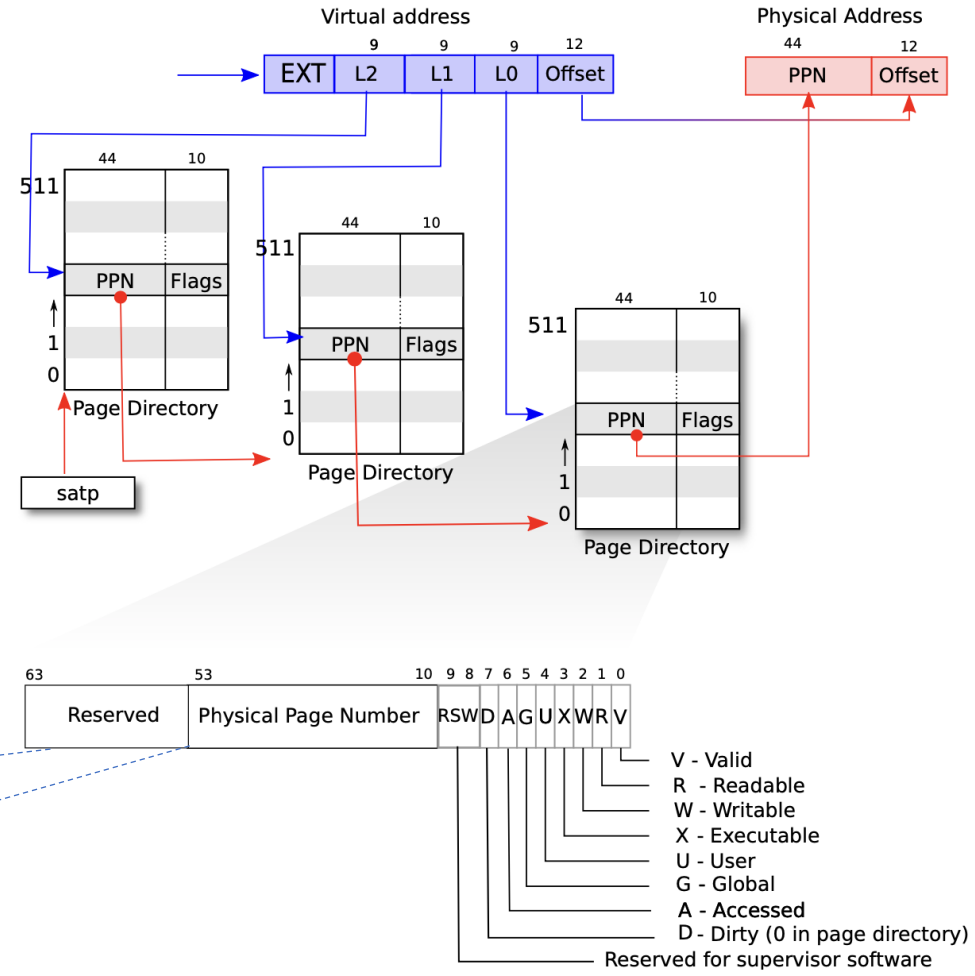
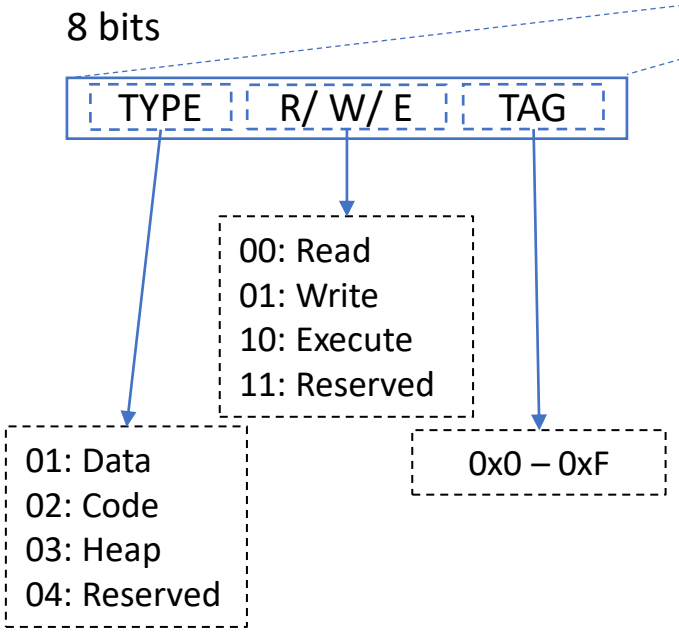
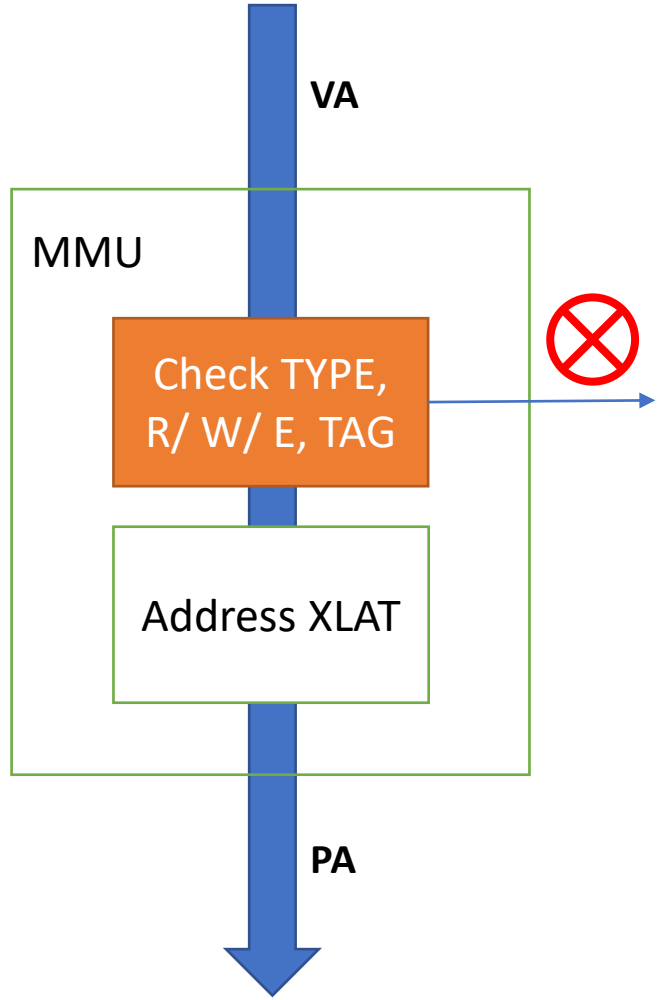


Figure 3.2: RISC-V address translation details.



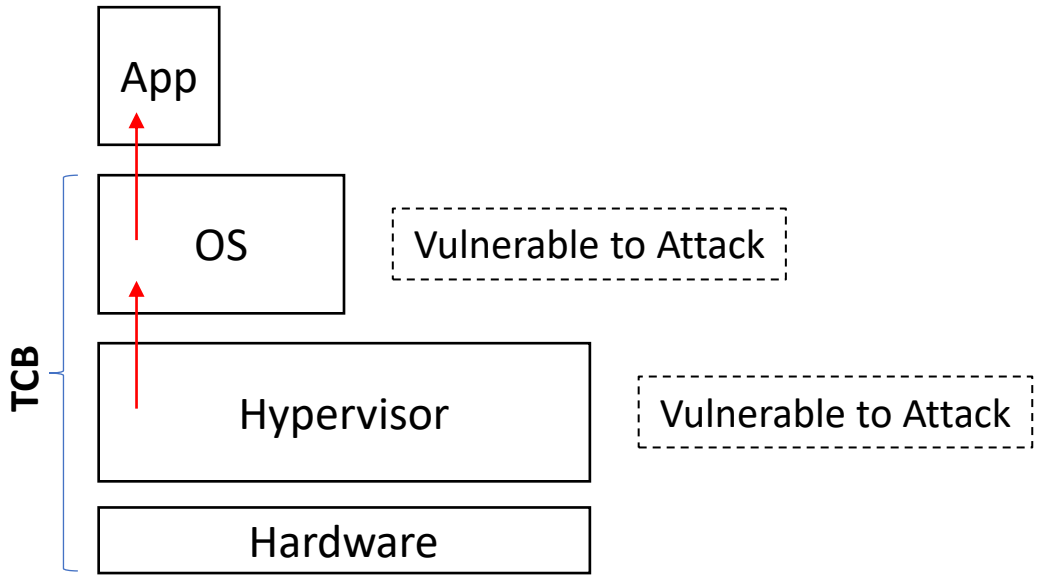


3. ENCRYPTED ISLANDS

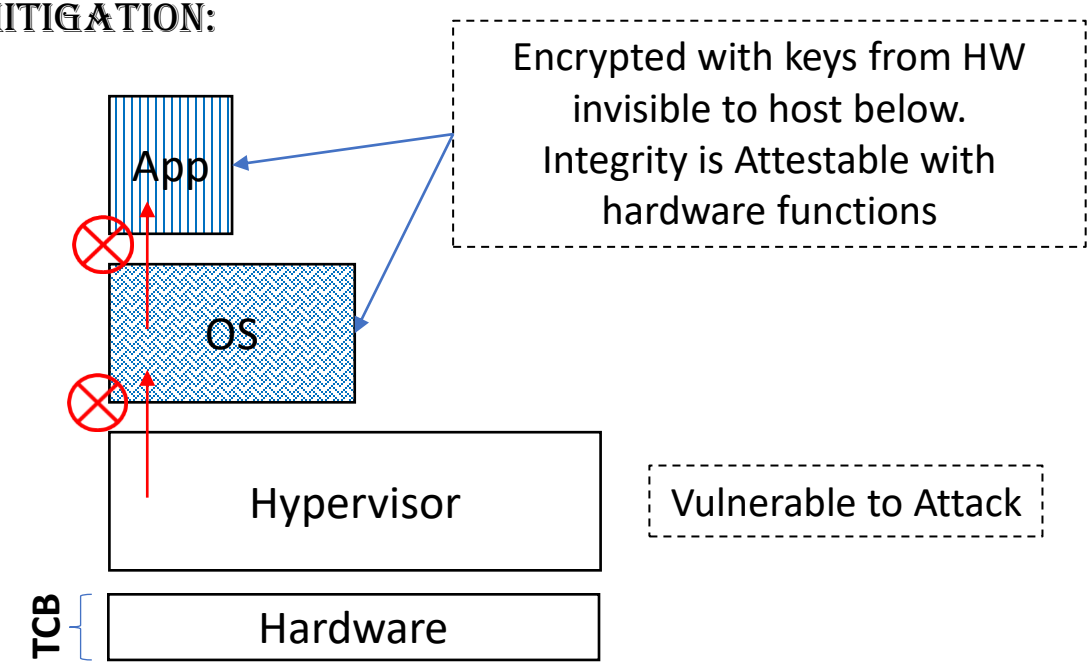
THREAT MODEL:

Asset	Description	Security Property C - Confidentiality I - Integrity A - Availability	Threat	Entry Point of Threat	Impact of Vulnerability	Severity (CVSS v3 Rating) https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator	Mitigation/ Security Requirement
Code/ Data	Software code and data	C	Disclose	Vulnerable OS or Hypervisor can be exploited with privilege escalation to view code/ data of application or hosted software	Disclosure of secrets	HIGH: 7.5 **CVSS v3.1 Vector** AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N	Encrypt code/ data via hardware mechanisms with hardware generated keys invisible to underlying OS or Hypervisor
Code/ Data	Software code and data	I	Tamper	Vulnerable OS or Hypervisor can be exploited with privilege escalation to tamper code/ data of application or hosted software	Compromised integrity of secrets	HIGH: 7.5 **CVSS v3.1 Vector** AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N	Integrity checking of code/ data by hardware that us attested by the hardware which can be verified remotely

ATTACK SCENARIO:



MITIGATION:



TBD