Análise de binários e sistemas assistida por *hardware*

Marcus Botacin Paulo de Geus André Grégio

XVIII SBSEG

2018

Roteiro

1. Introdução
2. Recursos e Características
3. Tecnologias
4. Técnicas
5. Aplicações
6. Desenvolvimentos Futuros
7. Conclusão

Roteiro

1 Introdução

2 Recursos e Características

3 Tecnologias

4 Técnicas

5 Aplicações

6 Desenvolvimentos Futuros

7 Conclusão

Quais aplicações falaremos neste curso ?

- Análise de *malware*.
- *Debugging*.
- Detecção de ataques.
- Verificações de integridade.
- Perícia forense.

Quais tecnologias falaremos neste curso ?

- *Hardware Virtual Machine* (HVM).
- *System Management Mode* (SMM).
- *Management Engine* (ME/AMT).
- *Hardware Performance Counters* (HPC).
- *Direct Memory Access* (DMA).
- *Software Guard Extensions* (SGX).

Este curso não é sobre:

- Programação SGX.
- Ataques de canais laterais (`Spectre`, `Meltdown` e outros).

O que eu preciso saber ?

- **Análise de *malware***: Traços de execução e *hooking* de chamadas de funções.
- ***Debugging***: *Breakpoints* e execução *step-by-step*.
- **Máquinas Virtuais**: *Host*, *guest* e *hypervisor*.
- **Sistemas Operacionais**: *Kernel*, *userland* e *syscalls*.
- **Arquitetura**: BIOS, Interrupções, DMA e MMU.
  - **Gerenciamento de memória**: Segmentos, paginação e *page faults*.
- **Programação**: *Callbacks*.
  - ***Assembly***: *Branches*.

Roteiro

1 Introdução

2 Recursos e Características

3 Tecnologias

4 Técnicas

5 Aplicações

6 Desenvolvimentos Futuros

7 Conclusão

Anti-Análise

1. **Verificação de integridade**.
2. **Identificação de efeitos colaterais de execução em ambientes instrumentados**.
3. **Identificação de variações nas medidas de tempo**.
4. ***Fingerprinting***.

Transparência

1. **Monitoração com privilégios superiores.**
2. **Execução livre de efeitos colaterais não privilegiados.**
3. **Instruções com semânticas idênticas ao caso base.**
4. **Tratamento transparente de exceções.**
5. **Medidas de tempo idênticas.**

Amplitude de monitoração

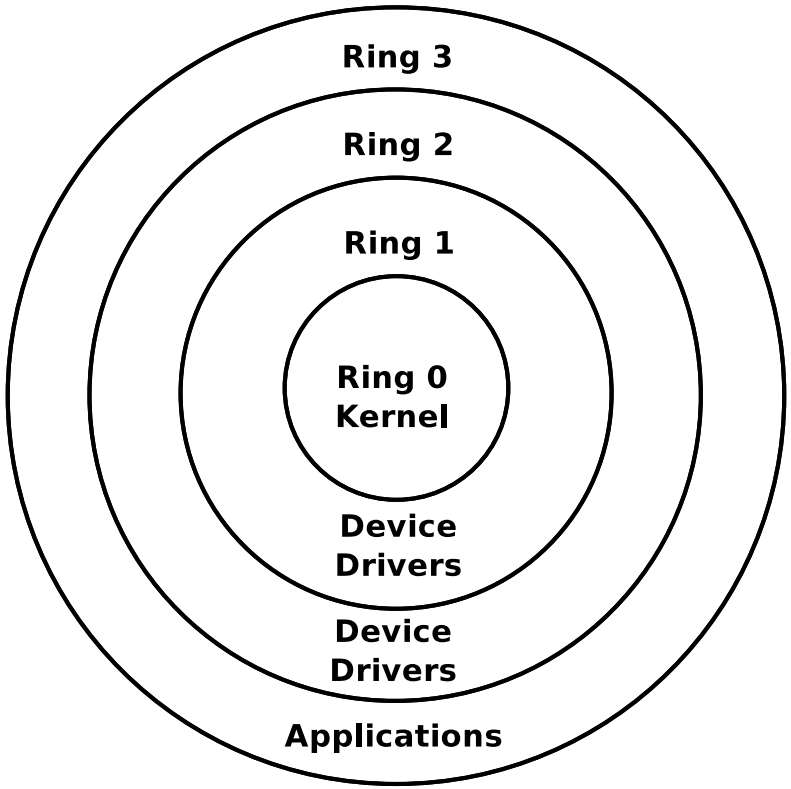

Figura: **Privilégios de Execução.** O *ring* 0 é o mais privilegiado e pode monitorar os demais *rings*. As aplicações executam em *ring* 3 e não podem interferir com os demais *rings*.

Amplitude de Monitoração

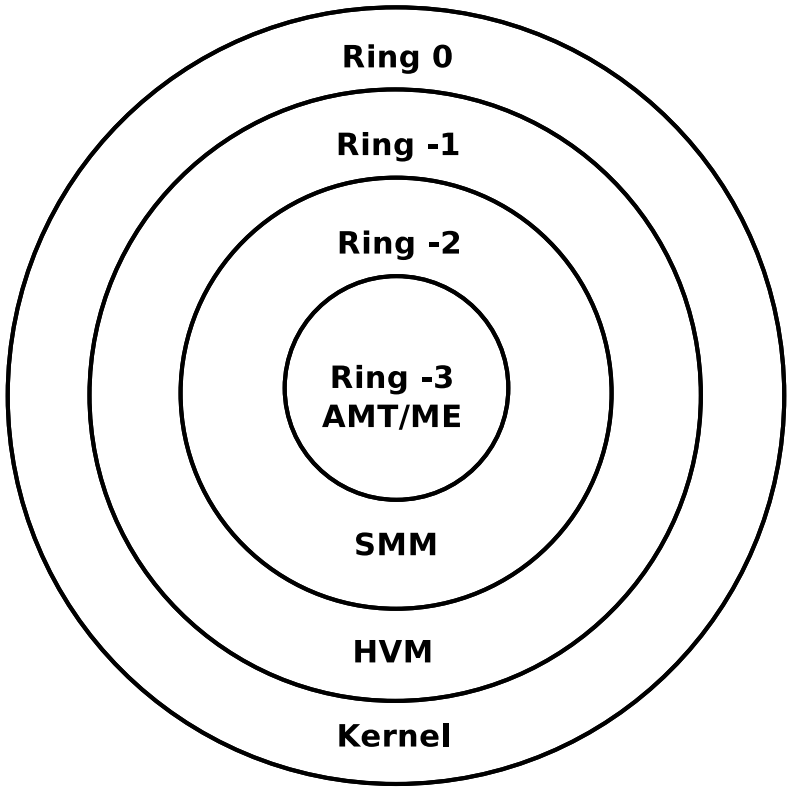

Figura: **Novos *rings* privilegiados.** O *ring* -3 é o mais privilegiado e pode monitorar os outros *rings*. Nesta nova configuração, o *kernel* executa dentro de uma máquina virtual de *hardware* (HVM), em *ring* -1.

Abstrações

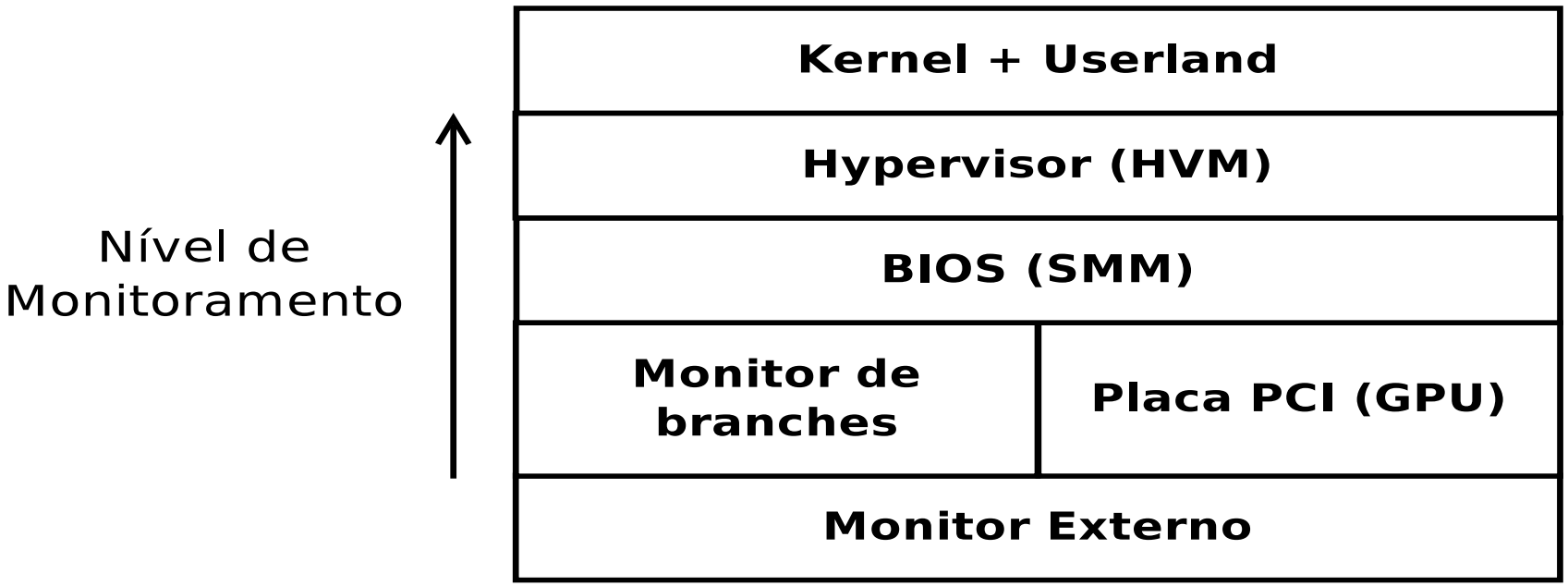

Figura: **Níveis de monitoramento.** Cada tecnologia apresentada neste capítulo monitora o sistema em um nível diferente. Os níveis mais altos se aproximam de soluções de *software*, enquanto níveis inferiores operam mais próximo do *hardware*.

Abstração

Figura: **Níveis de abstração.** Cada tecnologia apresentada neste capítulo opera em um nível de abstração diferente. Quanto mais alto o nível, mais próximo de uma informação compreensível para o ser humano. A passagem da informação de um nível de abstração para o outro exige procedimentos de introspecção para a adequação da sua representação.

Base de Código Confiável (TCB)

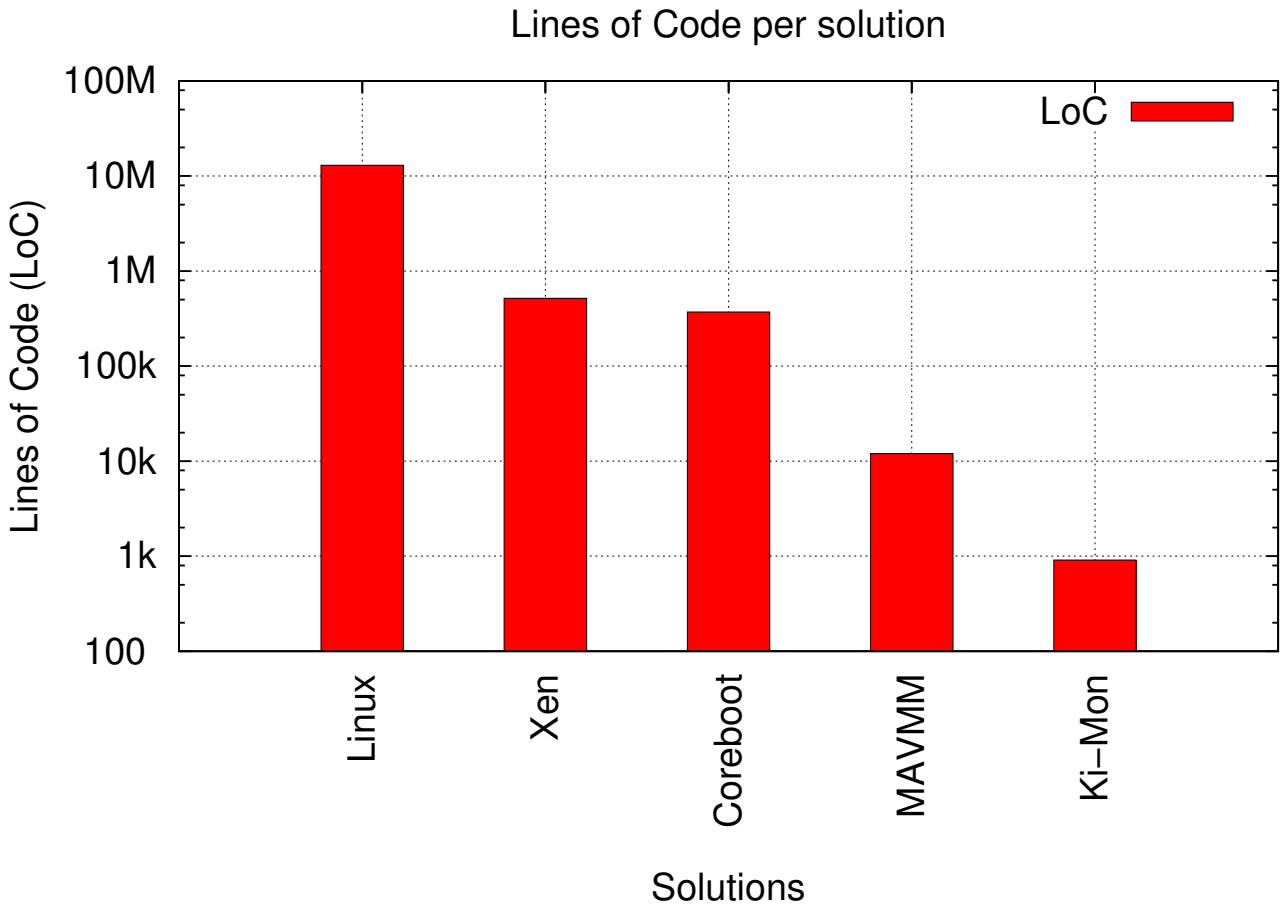

Figura: **Base de código confiável (TCB)**

Sincronia

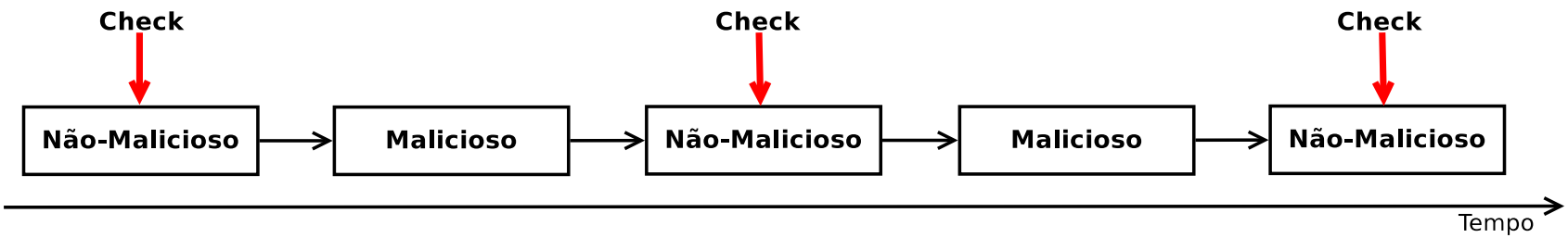

Figura: **Evasão de soluções do tipo *snapshot*.** Em soluções que implementem verificações síncronas, ações maliciosas podem não ser detectadas caso estas ocorram no intervalo entre duas inspeções.

Outras decisões de projeto

- **Posicionamento**: Interno x Externo.
- **Carregamento**: Tempo de *boot* ou tempo real.
- **Manutenção**: Verificações de integridade e consistência.
- **Desempenho**: Monitoração em tempo real ou análises *offline*.
- **Atualizações**: Dispensa da necessidade de recompilação (updates de assinaturas).
- **Integração**: Operação em sistemas legados e diminuição da curva de aprendizado.

Roteiro

1 Introdução

2 Recursos e Características

3 Tecnologias

4 Técnicas

5 Aplicações

6 Desenvolvimentos Futuros

7 Conclusão

Modos de operação dos processadores

Modos Originais

1. **Modo de endereçamento real**.
2. **Modo protegido**.
3. **Modo SMM**.

Modos Adicionais (HVM)

1. **Modo *VM root***.
2. **Modo *VM non-root***.

HVM

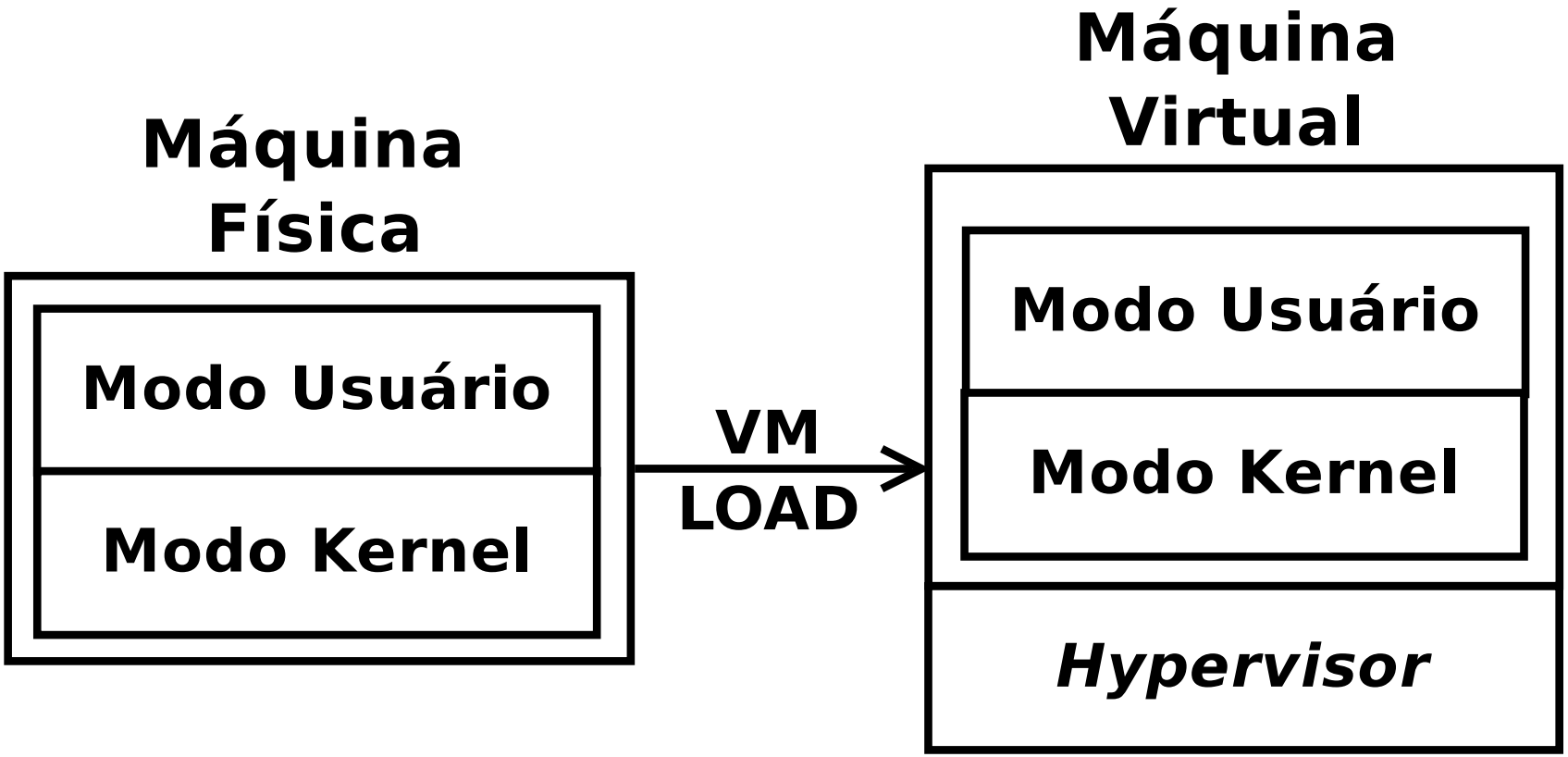

Figura: **Migração do sistema para máquinas virtuais.** Quando operando em uma máquina virtual, tanto o modo *kernel* quando o modo usuário ficam sobre supervisão de um *hypervisor*.

HVM

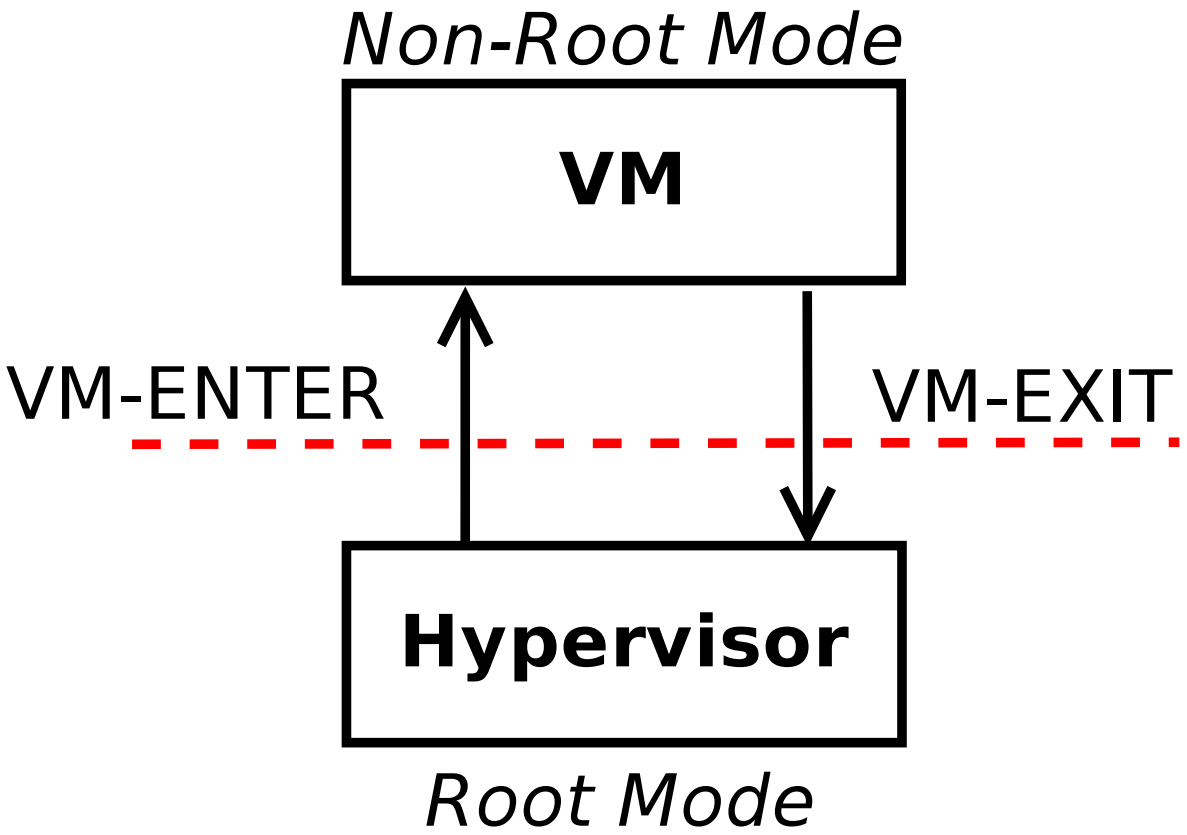

Figura: **Novos modos de operação.** A instrução `vmload` inicia a operação da máquina virtual. Quando uma ação monitorada ocorre no sistema *guest*, uma saída para o *hypervisor* é causada.

HVM

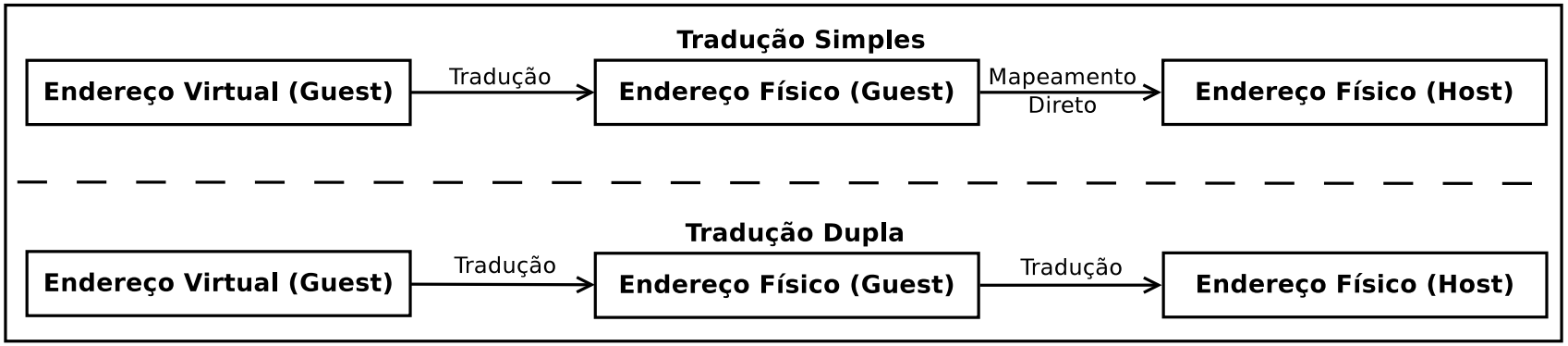

Figura: **Mecanismos de tradução de memória de um HVM.**
Enquanto máquinas virtuais por *software* apresentam apenas uma etapa de tradução, no *guest*, HVMs apresentam duas etapas, incluindo uma etapa adicional no *hypervisor*. Este mecanismo adicional de tradução pode ser instrumentado de modo a permitir o monitoramento de memória.

HVM

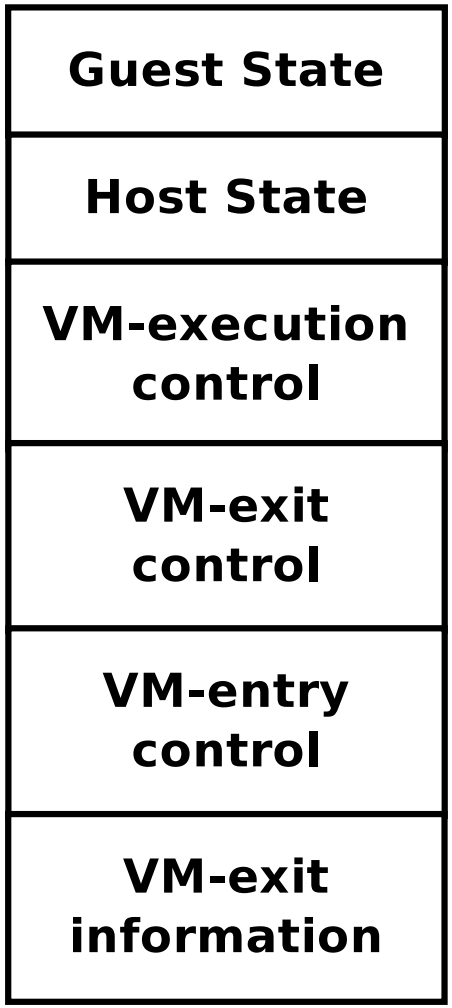

Figura: **Estrutura de Controle da máquina virtual. (VMCS)**

HVM

Código 1: Código do Xen. Eventos que causam saídas da máquina virtual.

```
static int vmx_init_vmcs_config(void){
    min = (CPU_BASED_HLT_EXITING |
           CPU_BASED_VIRTUAL_INTR_PENDING |
           CPU_BASED_CR8_LOAD_EXITING |
           CPU_BASED_CR8_STORE_EXITING |
           CPU_BASED_INVLPG_EXITING |
           CPU_BASED_CR3_LOAD_EXITING |
           CPU_BASED_CR3_STORE_EXITING |
           CPU_BASED_MONITOR_EXITING |
           CPU_BASED_MWAIT_EXITING |
           CPU_BASED_MOV_DR_EXITING |
           CPU_BASED_RDTSC_EXITING);
```

Modo SMM

- Gerenciamento de recursos (energia e temperatura).
- Execução em processador nativo.
- Código dentro da BIOS (sem *frameworks* de suporte).
- Não endereçada pelos demais modos.
- Acessível por interrupções (SMIs).

Modo SMM

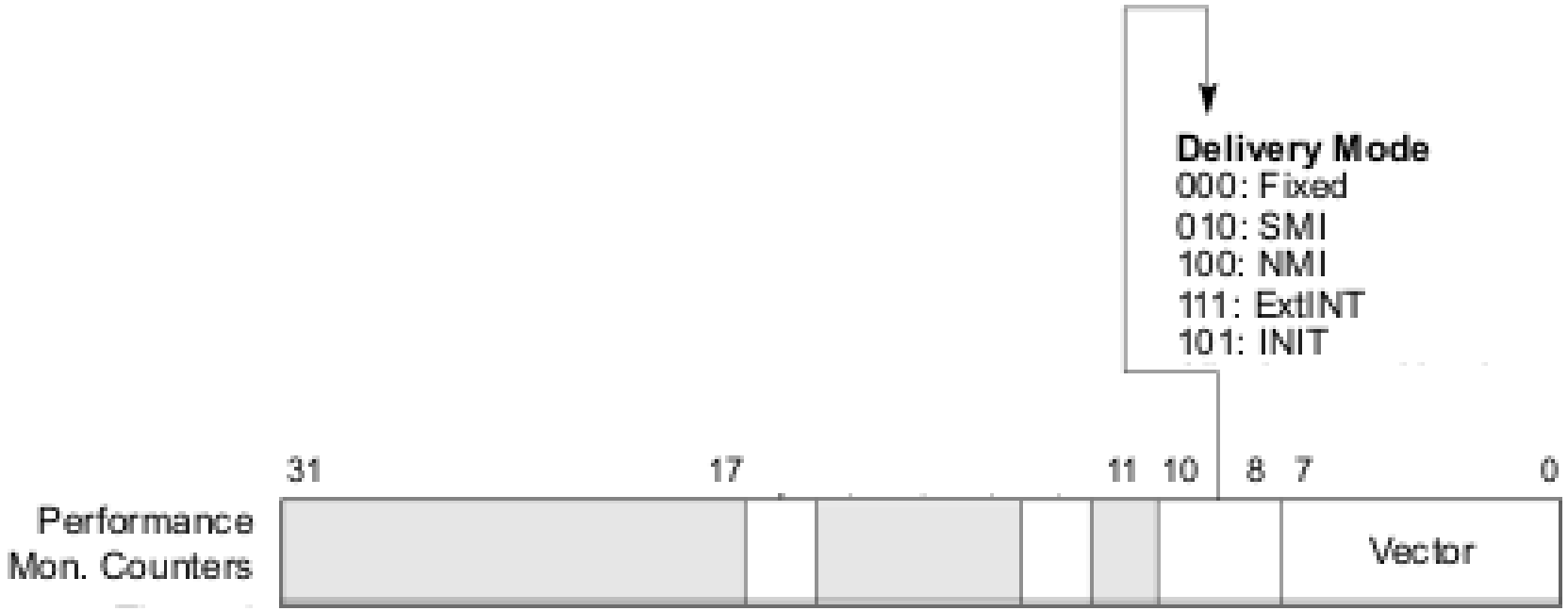

Figura: **Geração de SMIs.** Interrupções de diversos tipos, como as geradas pelos contadores de *performance*, podem ser entregues via SMIs e tratadas pelo modo SMM da BIOS.

Modo SMM

Código 2: Código do Coreboot. Tratamento de SMIs.

```
void smi_handler(u32 smm_revision) {
    unsigned int node;
    smm_state_save_area_t state_save;
    node=nodeid();
```

Modo ME

- Modo de gerenciamento.
- Controla os demais modos de operação.
- Localizado no *chipset*.
- Conta com ALU e DMA próprios.

Contadores de *performance* (HPC)

- Orientados a eventos.
- Operação como contadores ou monitores.
- Armazenamento em registradores ou páginas de memória.
- Interrupções quando atingem um *threshold*.
- Não isola processos.

Contadores de *performance* (HPC)

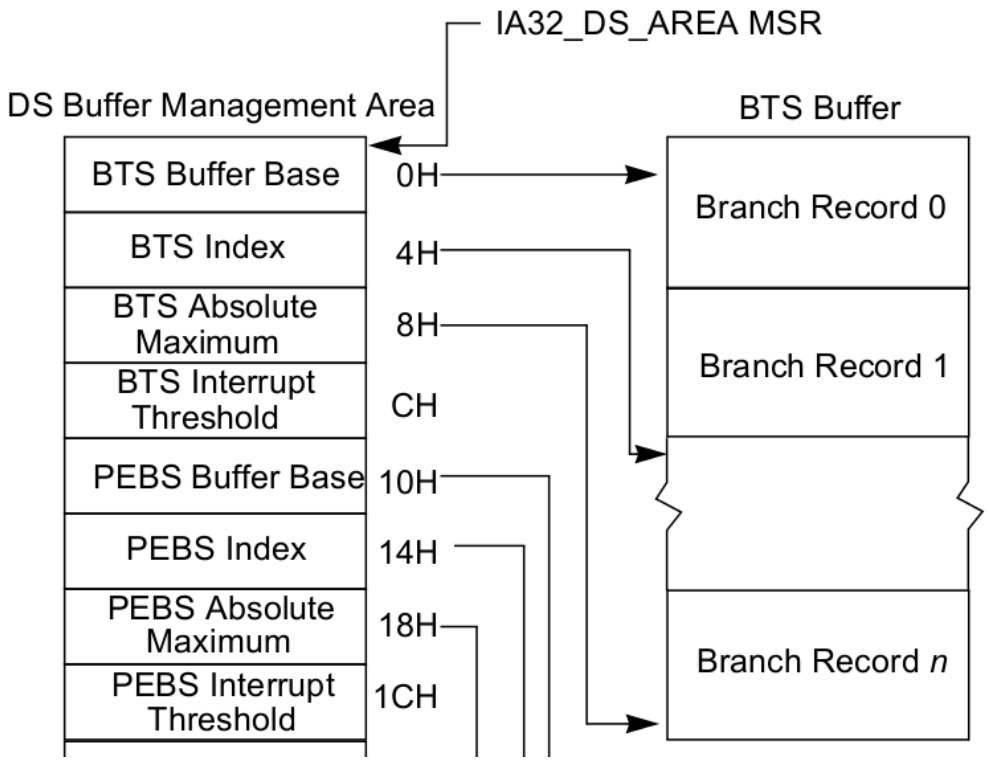

Figura: **Configuração do monitor de *branch*.** No monitor BTS, os dados são armazenados em páginas de memória. Quando o armazenamento atinge um *threshold* pré-estabelecido, uma interrupção é gerada.

Contadores de *performance* (HPC)

Bit Field	Bit Offset	Access	Description
CPL_EQ_0	0	R/W	When set, do not capture branches occurring in ring 0
CPL_NEQ_0	1	R/W	When set, do not capture branches occurring in ring >0
JCC	2	R/W	When set, do not capture conditional branches
NEAR_REL_CALL	3	R/W	When set, do not capture near relative calls
NEAR_IND_CALL	4	R/W	When set, do not capture near indirect calls
NEAR_RET	5	R/W	When set, do not capture near returns
NEAR_IND_JMP	6	R/W	When set, do not capture near indirect jumps
NEAR_REL_JMP	7	R/W	When set, do not capture near relative jumps
FAR_BRANCH	8	R/W	When set, do not capture far branches
Reserved	63:9		Must be zero

Figura: **Filtragem de eventos dos contadores de *performance*.** Os contadores de *performance*, dentre os quais os monitores de *branch*, podem filtrar os eventos monitorados por tipo (*branches* diretos, indiretos, chamadas de função) ou por nível de ocorrência (*kernel* ou modo usuário).

Enclaves Isolados

- Isolamento de páginas de memória pela MMU/TLB.
- As páginas são encriptadas nas trocas de contexto.
- As páginas são destruídas ao final da execução.
- *Hardware* criptográfico próprio.
- Compartilha cache e ALU.
- Exige recompilação.

Enclaves Isolados

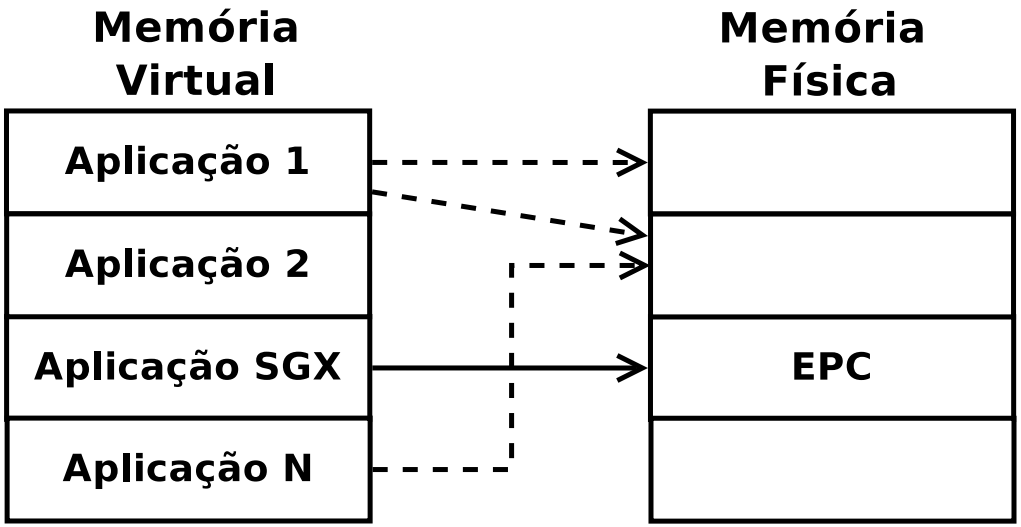

Figura: **Proteção de memória dos enclaves SGX**. O gerenciamento de memória dos enclaves (MMU e TLB) impede que outras aplicações mapeiem, direta ou indiretamente, a memória alocada para o enclave (*Enclave Page Cache*—EPC).

Acesso Direto à Memória (DMA)

- Forma de não bloquear a CPU com tarefas secondárias.
- Forma frequente de transferir dados entre placas PCI e seus *buffers*.
- Sem controle de permissões da MMU.
- Inserção de placas PCI não é autenticada.

Acesso Direto à Memória (DMA)

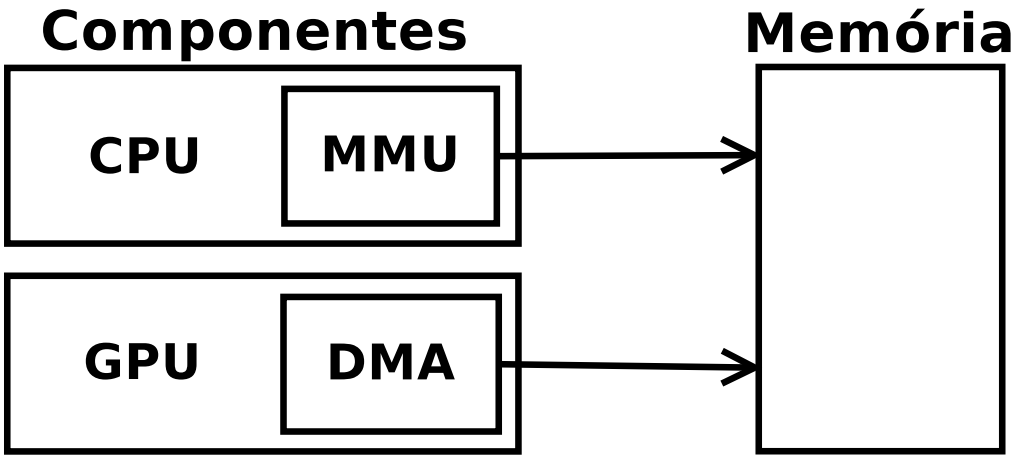

Figura: **Acesso direto à memória**. Este recurso permite que placas PCI, tais como GPU, acessem a memória diretamente. Como o acesso não é protegido pela MMU ou outro componente, os dispositivos PCI podem mapear a memória do sistema como um todo, incluindo as mesmas regiões acessadas pela CPU. A enumeração da memória como um todo permite, por exemplo, o *dump* para fins de forense.

Componentes Externos

- Co-processadores.
- Compartilham barramentos.
- *Tamper-proof*.
- Componentes passivos.
- Introspecção sem valores de registradores.

Componentes Externos

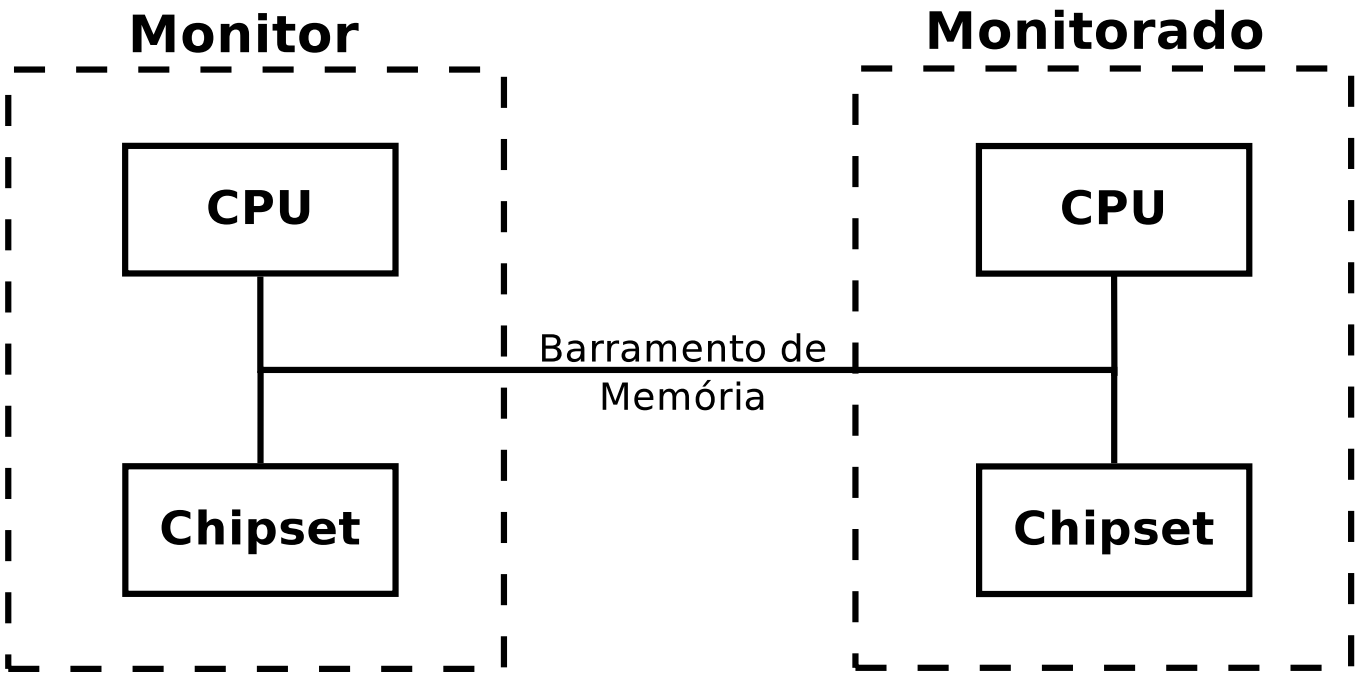

Figura: **Monitoração com *hardware* externo**. Dispositivos externos podem ser conectados a um barramento de memória compartilhado com a plataforma monitorada para realizar *snooping* dos dados.

Roteiro

1 Introdução

2 Recursos e Características

3 Tecnologias

4 Técnicas

5 Aplicações

6 Desenvolvimentos Futuros

7 Conclusão

Ações de Interesse

- **Eventos**: Gatilhos baseados em mecanismos já existentes no sistema (VM-Exits).
- **Callbacks**: Gatilhos baseados em código inserido pela solução de monitoração.

Monitoramento de Memória

- **Controles e Proteções da MMU**: Presença, Leitura, Escrita e Execução.
- **Violações**: Exceções e *faults*.

Monitoramento de Memória

Write Xor Execute

Código 3: Pilha não executável.

```
cat /proc/self/maps
00400000-0040c000 r-xp 00000000 /bin/cat
7f0bef204000-7f0bef3c4000 r-xp   libc-2.23.so
7ffe3a213000-7ffe3a234000 rw-p   [stack]
```

Monitoramento de Memória

SMC

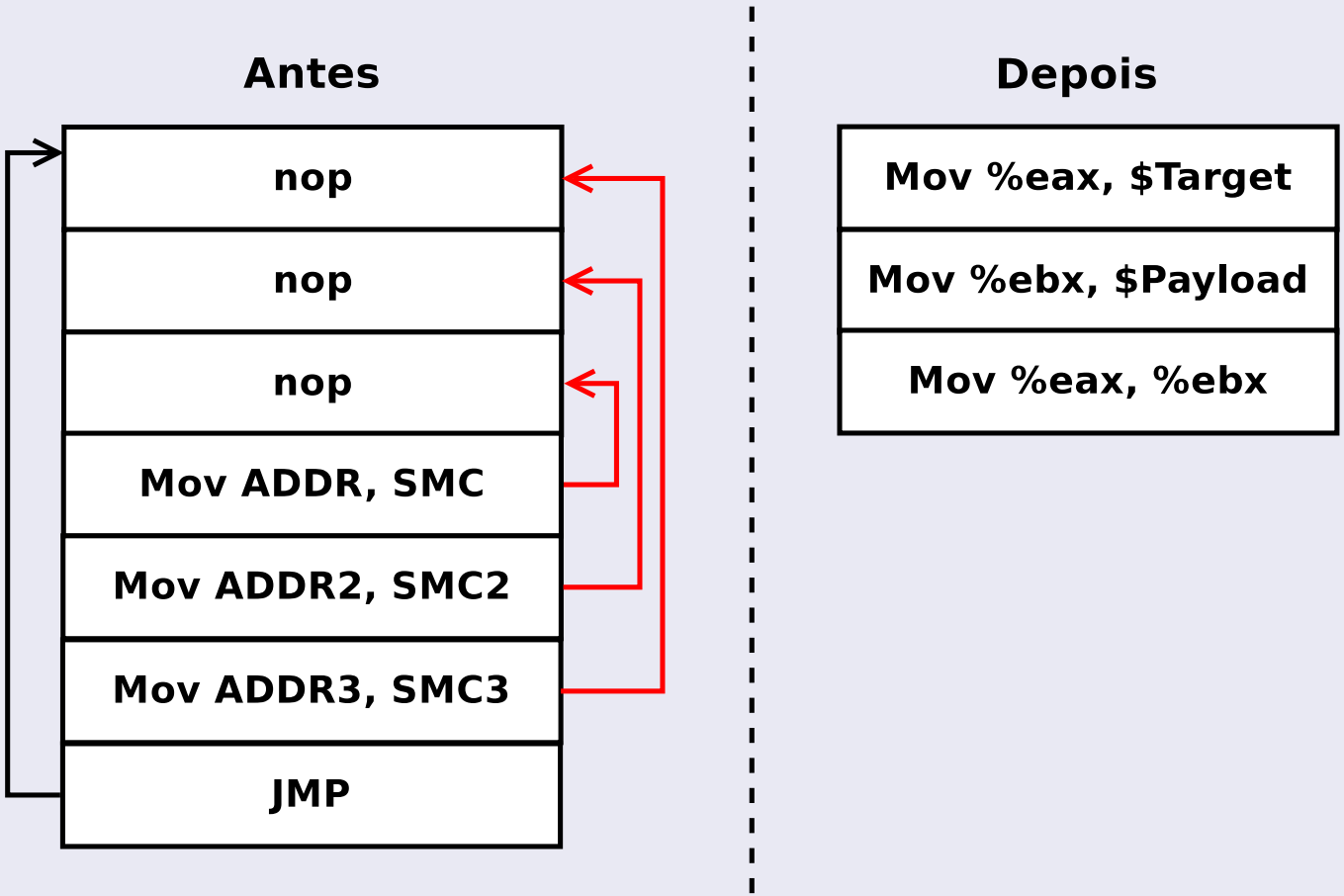

Figura: **Código Auto-Modificável (SMC)**

Monitoramento de Memória

- ***Dump***: Cópia da memória para análise.
- **Unix**: *Copy On Write*.

Monitoramento de Memória

Shadow Copy

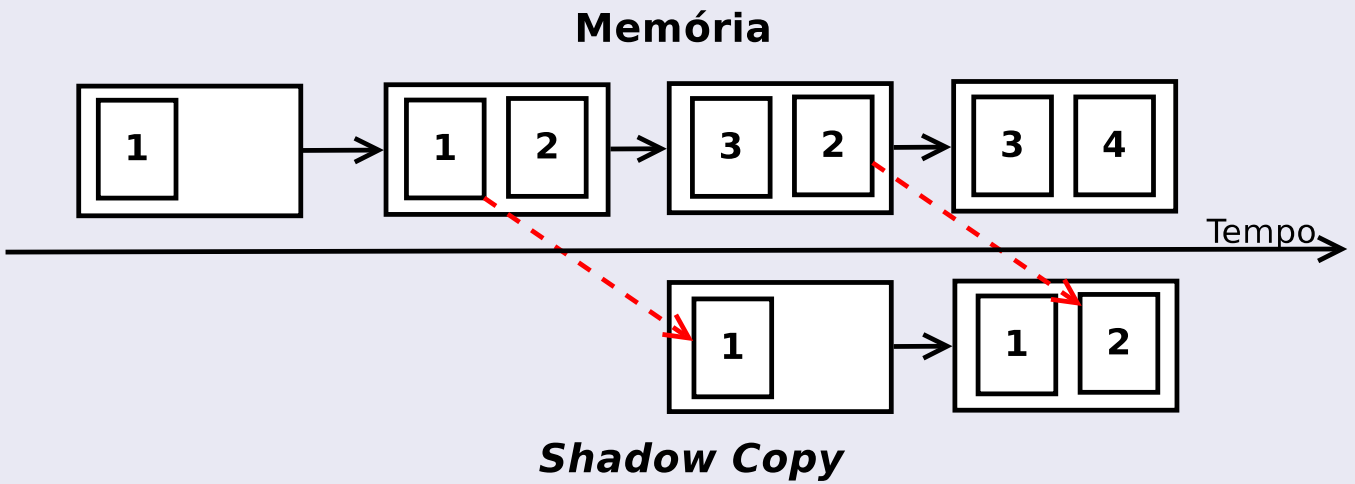

Figura: *Shadow Copies*

Controles de Granularidade

- **Pontos de parada**: *Breakpoints* a partir de *faults* e *overflows*.
- **Instruções Individuais**: Suporte de monitores de *branch* para introspecção.
- ***Step by Step***: *Step-Into* e *Step-Over*.

Controle de Granularidade

Step-Into

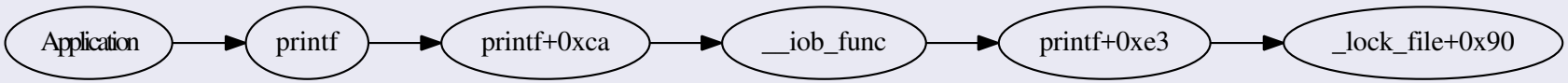

Figura: *Step-Into.* Mergulho nas funções internas.

Step-Over

Figura: *Step-Over.* Funções internas são desconsideradas.

Mitigações

- ***Fingerprint***: Técnicas de *rootkit* para esconder os *drivers* de carregamento.
- **Evasão por tempo**: Adulteração das medidas do *TimeStamp Counter* (TSC).

Roteiro

1 Introdução

2 Recursos e Características

3 Tecnologias

4 Técnicas

5 Aplicações

6 Desenvolvimentos Futuros

7 Conclusão

Detecção de *bugs*

Código 4: Desvios Tomados.
Endereços dos desvios.

```
je      4b9cf6
jne     4b9c46
jle     4b9bd3
jne     4b9c9c
je      4b9c9c
jne     4b9c7a
je      4b9ce8
je      4b9ae3
```

Detecção de *bugs*

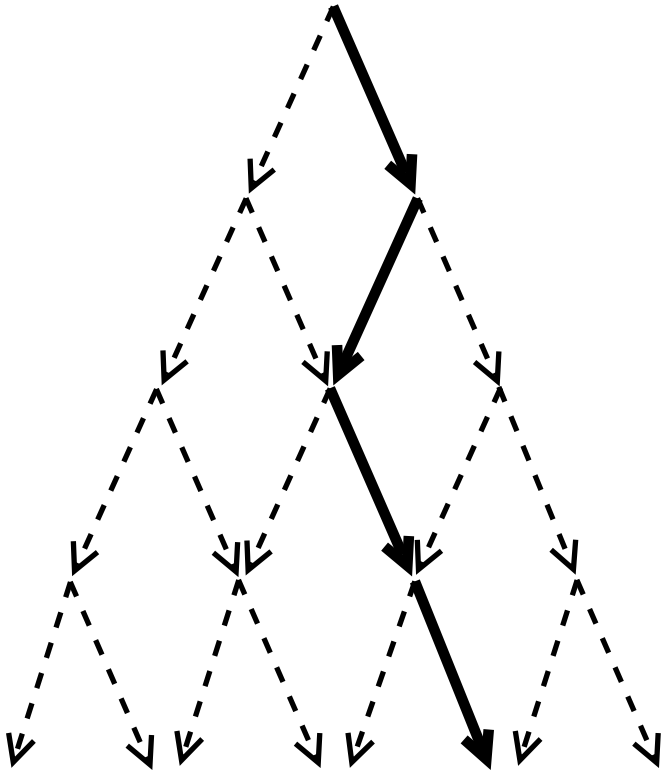

Figura: **Árvore de decisão.** O acompanhamento dos *branches* tomados permite identificar o caminho percorrido e, assim, identificar em qual ponto a execução diverge do esperado.

Aplicações

- **Análise de *malware***: Realizada de forma transparente, superando técnicas de anti-análise.
- ***Debugging***: Contando com a visão de todo do sistema, permitindo depurar o *kernel*.
- **Imposição de políticas**: Beneficiando-se da interposição de eventos (e.g., criptografia de I/O).

Forense

- **Carregamento em tempo real**: *Late Launch* de HVM.
- **Carregamento em sistemas comprometidos**: Detector de mentiras (*lie detector*).

Forense

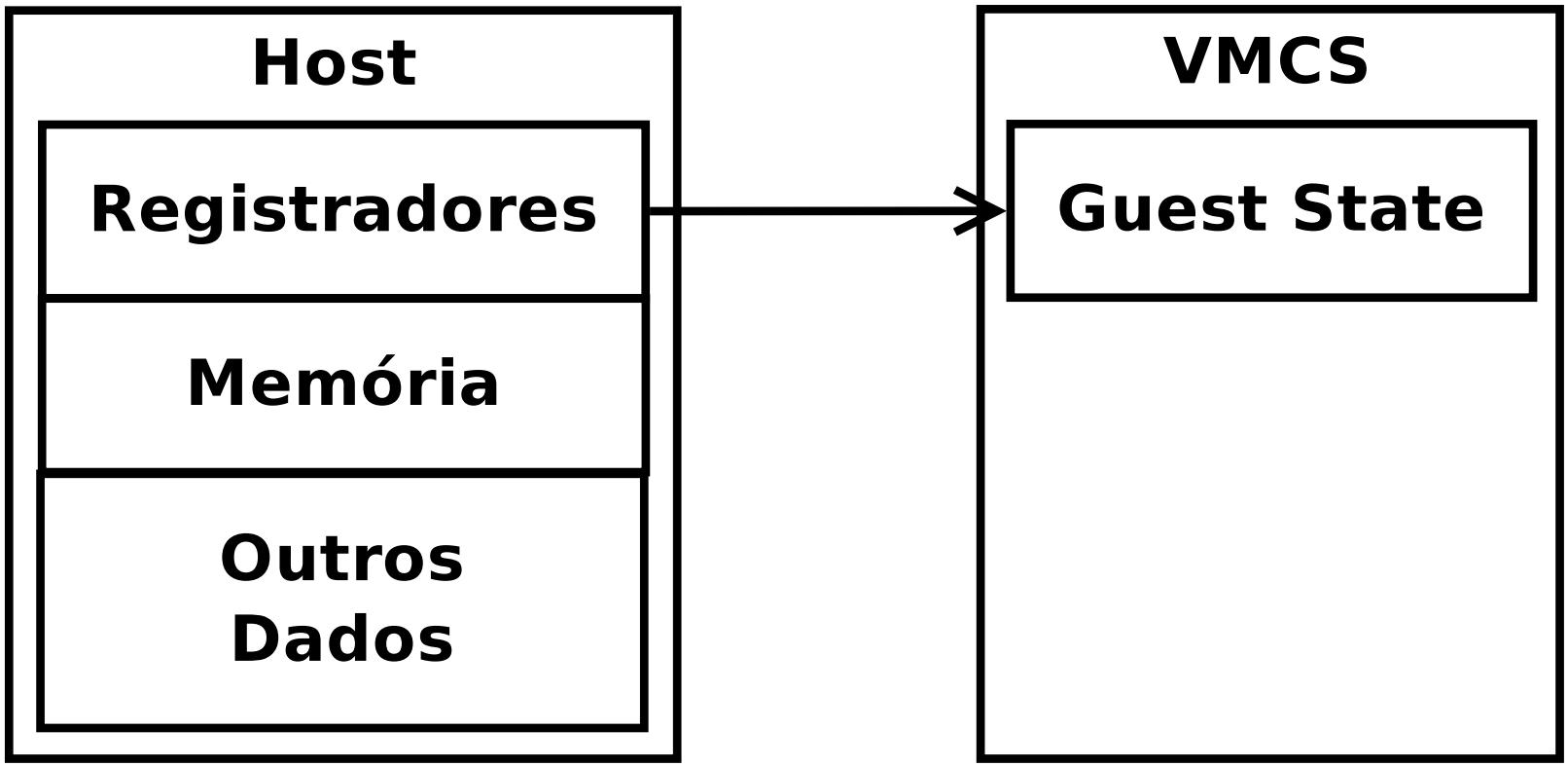

Figura: **Carregamento em tempo real.**

Detecção de Ataques

ROP

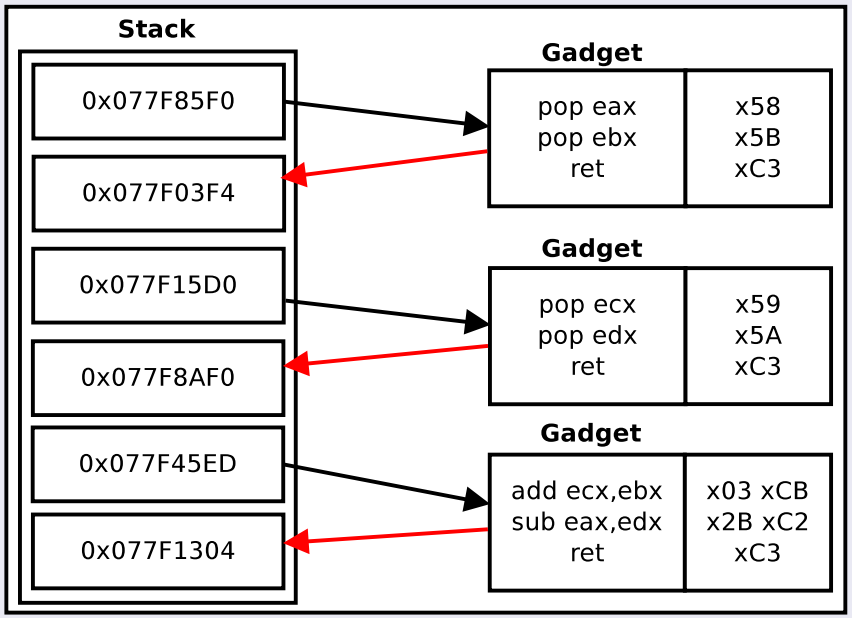

Figura: **Programação Orientada a Retorno (ROP).** O *payload* neste tipo de ataque é construído através do encadeamento de sequências de instruções terminadas por uma instrução de retorno (*gadgets*).

Detecção de Ataques

CFI

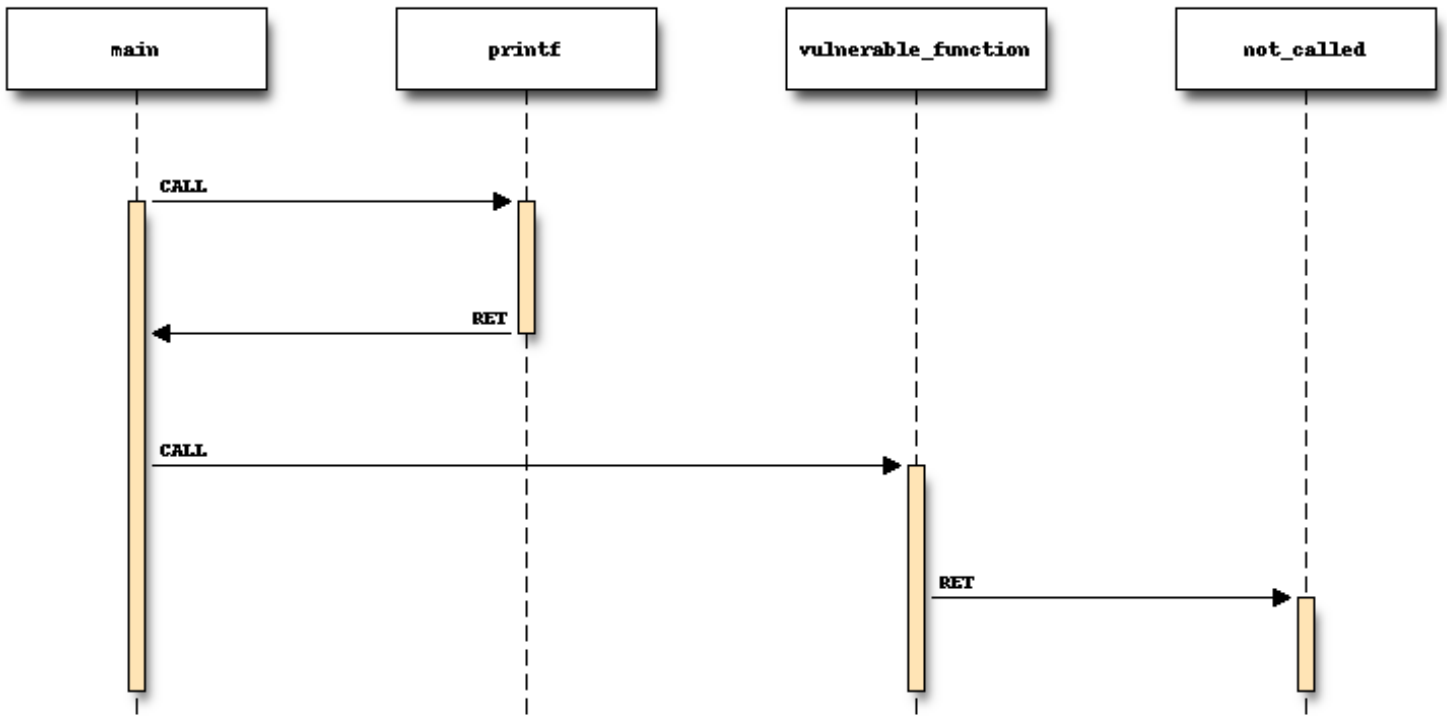

Figura: **Integridade de Fluxo de Controle (CFI).**

Verificação de Integridade

Ataques ao *hypervisor*

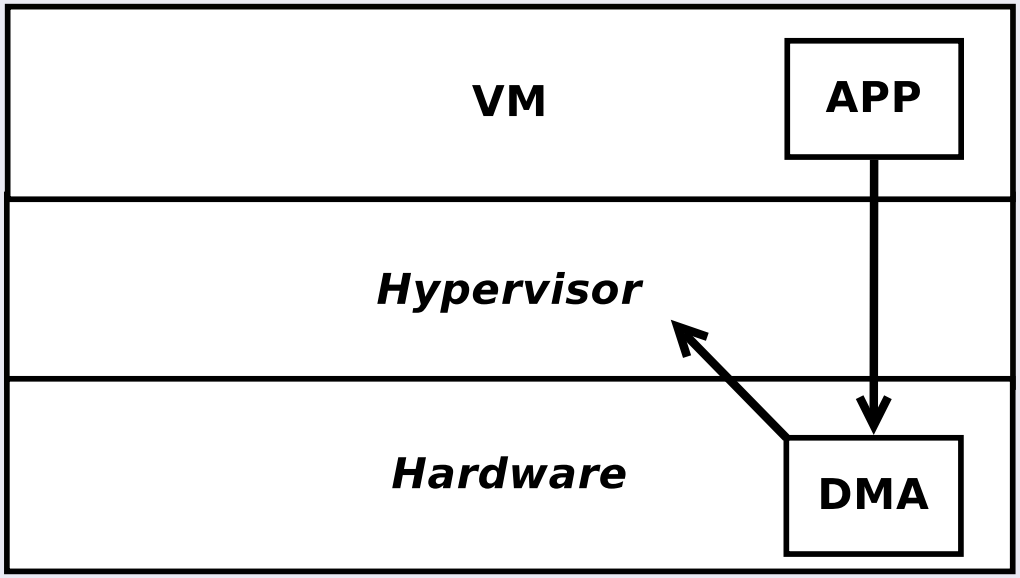

Figura: Ataques ao *Hypervisor*. A aplicação dentro da máquina virtual pode mapear a memória física do *hypevisor* e, através de acesso DMA, modificar seu conteúdo. A aplicação pode, por exemplo, alterar as permissões que o *hypervisor* atribui a sua execução, elevando, assim, seus privilégios.

Roteiro

1 Introdução

2 Recursos e Características

3 Tecnologias

4 Técnicas

5 Aplicações

6 Desenvolvimentos Futuros

7 Conclusão

HVMs

Hypercalls

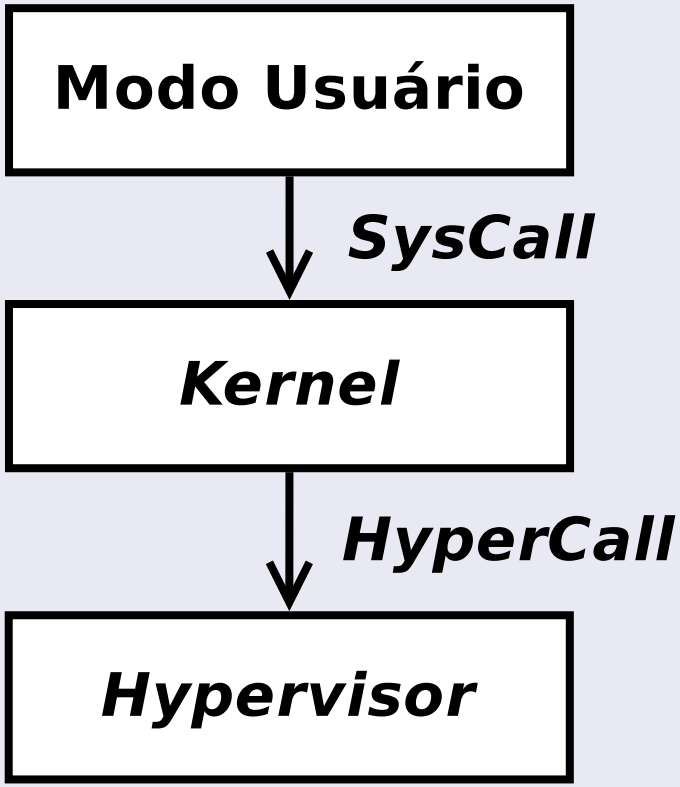

Figura: *Hypercall.* Chamadas do *hypervisor* a partir do *kernel* são correspondentes a chamadas deste pelo modo usuário.

HVM

Hypercalls

Código 5: Código do Xen. Hypercalls já existentes.

```
enum hypercall_num {
    #define __HYPERVISOR_set_callbacks
    #define __HYPERVISOR_set_debugreg
    #define __HYPERVISOR_get_debugreg
    #define __HYPERVISOR_xen_version
    #define __HYPERVISOR_vm_assist
    #define __HYPERVISOR_callback_op
    #define __HYPERVISOR_sysctl
    #define __HYPERVISOR_domctl
```

Oportunidades e Desafios

- **HVM**: Virtualização aninhada.
- **Modo SMM**: Superar *gap* semântico para inspecionar HVMs.
- **Modo ME**: Desenvolvimento de analisadores.
- **Contadores de performance**: Expansão baseada nos avanços em aprendizado de máquina.
- **DMA**: Bloqueio de dispositivos maliciosos.
- **Hardware Externo**: Desenvolvimento de soluções ativas.

Novas Ameaças

- **Modo SMM**: *Stealth Man-In-The-Middle*.
- **DMA**: *Keyloggers*.
- **Modo ME**: *Rootkits* em modo *system-wide*.
- **SGX**: *Malware* em execução dentro do enclave.

Enclaves Isolados

Malware em SGX

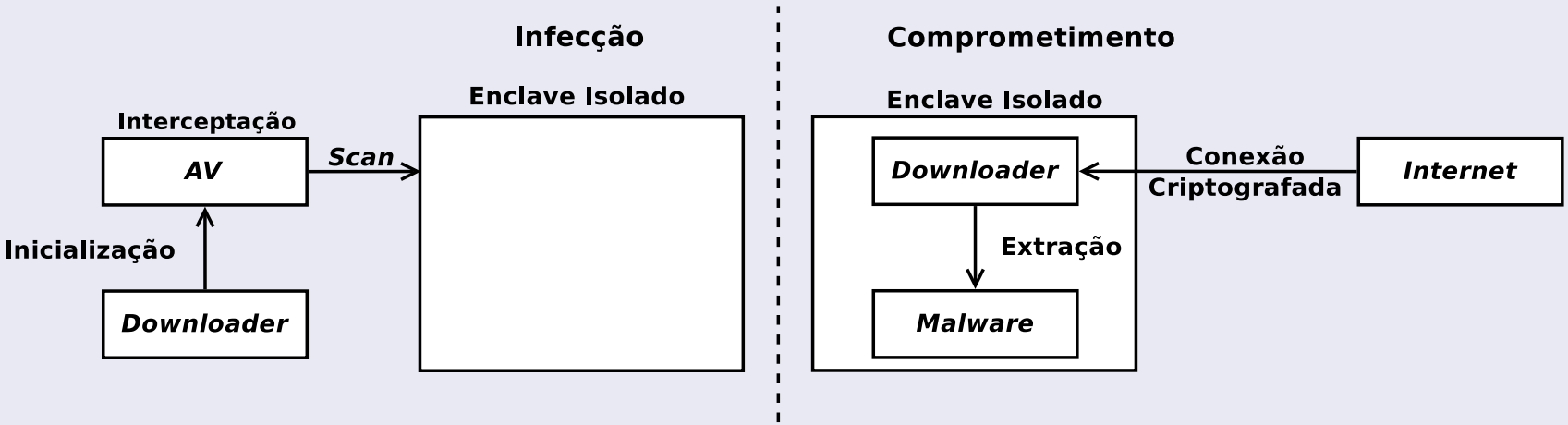

Figura: *Malware* em um enclave isolado.

Roteiro

1 Introdução

2 Recursos e Características

3 Tecnologias

4 Técnicas

5 Aplicações

6 Desenvolvimentos Futuros

7 Conclusão

Sumário

- Aplicações e plataformas modernas e complexas sofrem ameaças modernas e complexas.
- O suporte de *hardware* pode auxiliar no desenvolvimento de mecanismos de monitoração.
- O mesmo suporte pode ser usado para implementar novas ameaças mais efetivos e eficientes.

Dúvidas & Sugestões ?

Contato

mfbotacin@inf.ufpr.br