Control Flow Enforcement Technology (CET)

Compiler Architecture and Tools Conference (CATC) 2017

ittai.anati@intel.com

oren.ben.simhon@intel.com

December 2017

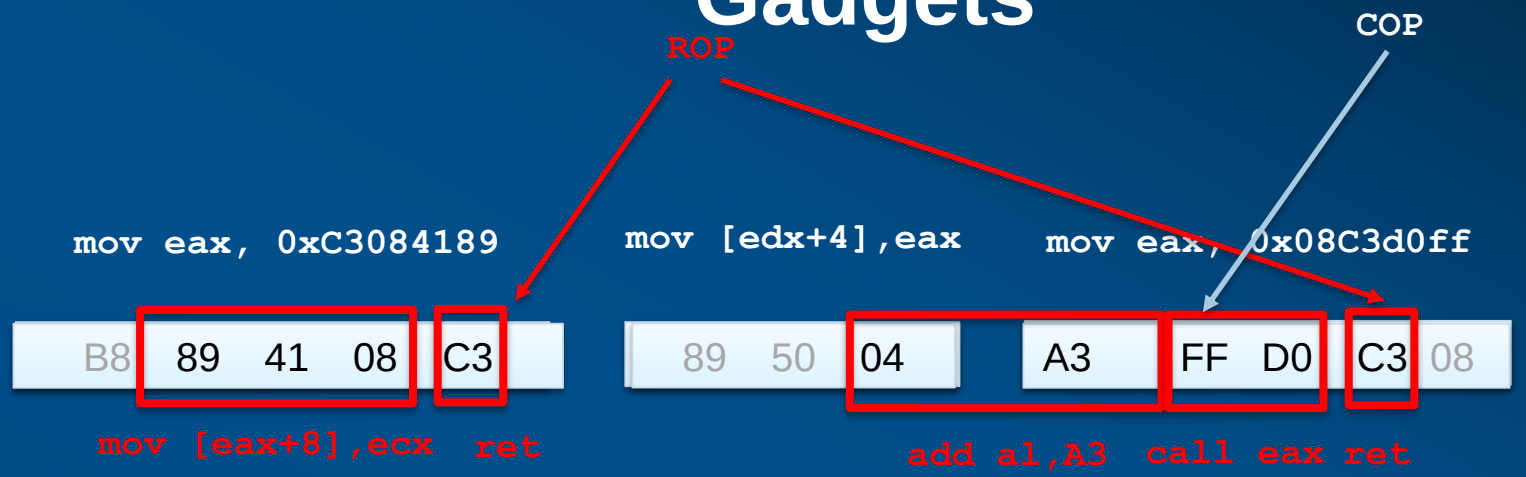
Disclaimers

- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. Learn more at Intel.com, or from the OEM or retailer.
- You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.
- No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.
- Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.
- No computer system can be absolutely secure.
- Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.
-

Agenda

- Introduction
- Motivation
- Technology overview
  - Shadow Stack
  - Endbranch
- LLVM Compiler enabling
- Summary

Gadgets



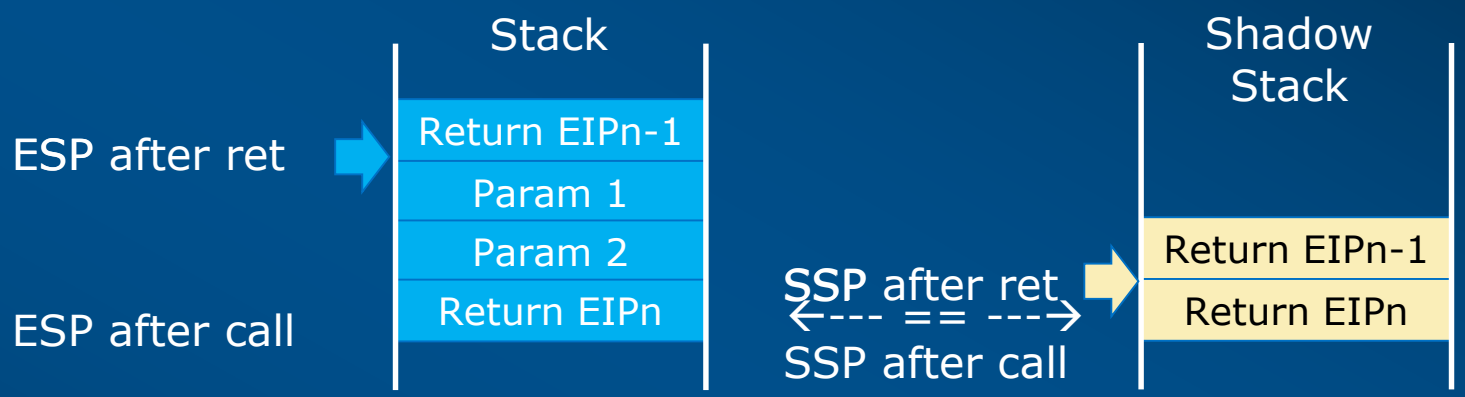
- Modern CPU enhancements prevent code injection: Non-executable stack and non-writeable code pages
- However: IA allows instruction decoding to start from any byte, providing attackers with a different set of instructions than intended by the programmer
- Attackers scan the code for meaningful snippets (gadgets) and chain them together through:
  - ROP – “Return” oriented programming
  - JOP – “Jump” oriented programming
  - COP – “Call” oriented programming

Intel's Control-flow Enforcement Technology - CET

- Goals
  - SW friendly ISA to protect from ROP/COP/JOP
  - Comprehensive solution with minimal impact to application developers
  - Broad enabling via OS, Dev Tools and Runtime
  - Acceptable performance and impact on energy usages
- Architecture
  - Two mechanisms to enhance protection against unintended changes to execution flow
  - ROP: SHADOW-STACK for protecting return addresses
  - JOP/COP: ENDBRANCH instruction for marking legal target addresses of indirect jumps and calls
  - Each mechanism can be enabled separately per privilege level

SHADOW STACK

Shadow Stack Operation



Stack usage on near CALL

SHADOW STACK

- Setup by OS/VMM
- Protected by new memory access control
- Different shadow stacks for each privilege level

- Call
  - pushes return address on both stacks
- No parameters passing on shadow stack

- Return
  - pops return address from both stacks
  - Controlflow Protection (#CP) exception in case the two return addresses don't match

Keeps stack ABI intact – no changes to data stack layout

Managing the Shadow Stack

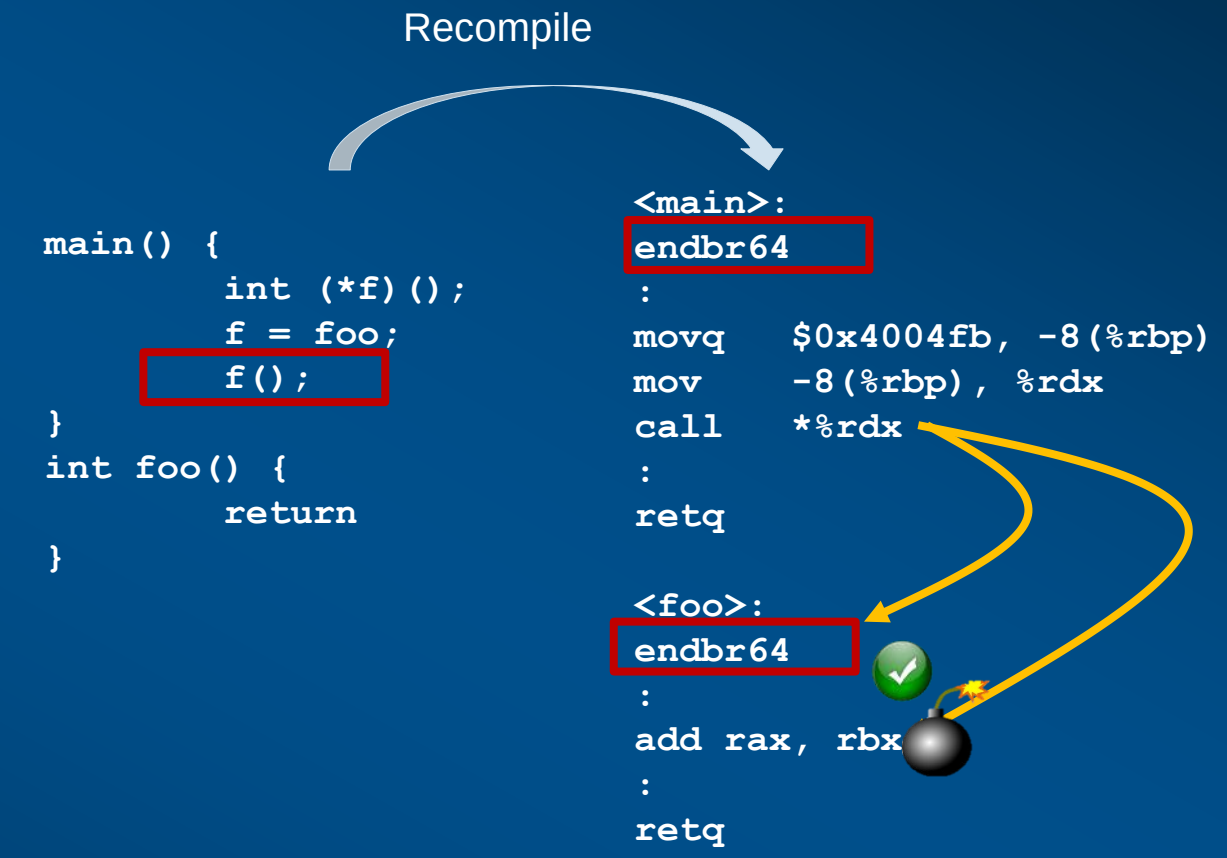
- OS/VMM sets up the Shadow stack for the application
- Some usages require to manage the Shadow Stack pointer
  - Stack unwinding
  - User mode thread switching
- New instructions help manage shadow stack securely:
  - RDSSP + INCSSP – To set checkpoint and unwind stack
  - SAVEPREVSSP/RSTORSSP – save/restore shadow stack pointer for user mode thread switching
- Full list in spec

Minimal changes to securely support common software constructs

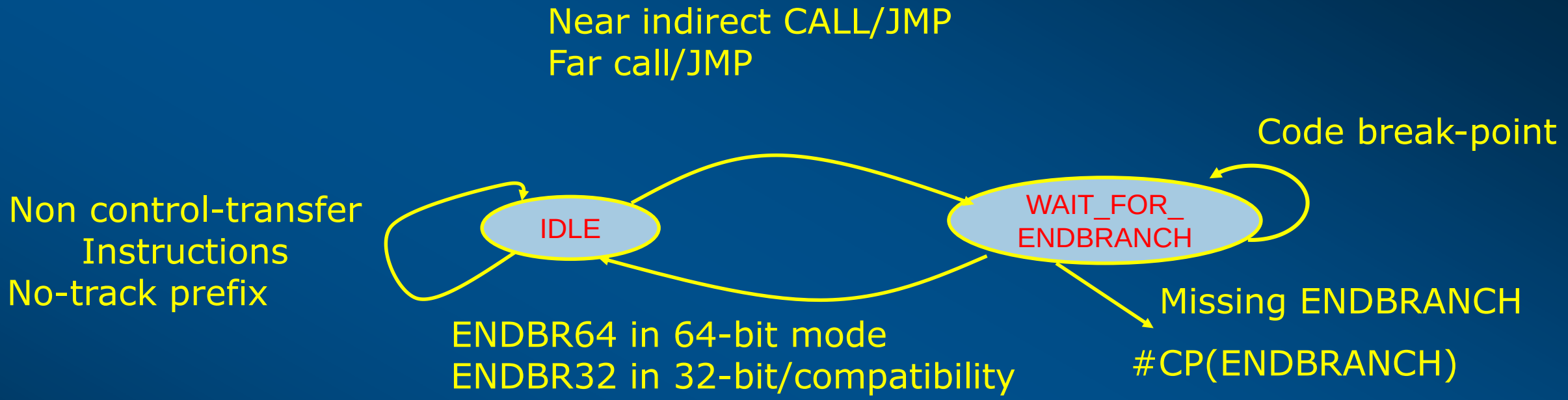
ENDBRANCH

ENDBRANCH

- New Instruction to mark legal targets of indirect jumps
- Added by the compiler
- Decodes as “NOP” on legacy processors
- An indirect jump to a target not marked by ENDBR signals an exception



CET ENDBRANCH State Machine



- Separate state machines for user and for supervisor mode
- Special no-track prefix for protected jumps to reduce ENDBRANCH footprint
- Special treatment for debugging (INT3 opcode “CC”)

Compatibility with legacy libraries

- Legacy libraries don’t have ENDBRANCH and could crash the application
- OS/VMM can set the legacy compatibility treatment bitmap for the context
  - If the bitmap indicates page has endbranch-enabled code, #CP exception is signaled
  - If the bitmap indicates page has legacy code, the violation is waived
- OS can chose between a one-time waiver and suspending endbranch violations until an ENDBR32/64 is detected
- Legacy compatibility treatment could potentially greatly impact performance, so users are encouraged to use recompiled libraries

COMPILER SUPPORT

ABI Changes

- Updated System V ABI with Intel CET extension
- Program loader Updates
  - Enables Shadow-Stack (SHSTK) if the executable and all shared objects are SHSTK enabled
  - Enables Indirect Branch Tracking (IBT) if the executable is IBT enabled and mark non-IBT enabled shared objects as legacy using an allocated bitmap
- The linker creates IBT-enabled PLT

Compiler CET Considerations

- Compilers minimize the use of ENDBR instruction in order to:
  - Avoid possible attacker landing pads
  - Reduce code size (consider huge switch cases)
- For guarded (range check) switch cases, ENDBR instructions are not added
- The compiler doesn't always know which function will be called by indirect call
- Compiler provides 'no_track' attribute:
  - Function: To avoid 'ENDBR' at the beginning of a function
  - Function Pointer: To add 'no_track' prefix before an indirect call
- Compiler updates exceptions handling builtins to fix the shadow stack
  - For example: To fix shadow stack pointer after SetJmp/LongJmp

Compiler Status

- GCC/LLVM/ICC/MS compilers have started implementing CET support
- GCC team updates GNU libraries, loader and linker
- LLVM is currently in the stage of open-sourcing CET support

SUMMARY

Summary

- Intel's Control-Flow Enforcement Technology (CET) provides a comprehensive solution to enhance protection against ROP/JOP/COP attacks
  - SHADOW STACK: Enhanced protection against ROP attacks
  - ENDBRANCH: Enhanced protection against JOP/COP attacks
- Protects applications and supervisor code
- Minimal impact to existing SW and to application developers

- Spec is published, search for intel CET