Practical Fine-Grained Binary Code Randomization

Soumyakant Priyadarshan, Huan Nguyen, R. Sekar
SecLab

Stony Brook University
Computer Science

Static attack	Direct-disclosure attack	Indirect-disclosure attack
Attacker has a copy of the victim's binary	Attacker discloses victim's code memory	Attacker discloses victim's data memory

Drawbacks of Existing Approaches

Require Source Code: Incompatible with dominant software deployment and update mechanisms.
Poor Performance: Previous binary-based techniques have high overhead.
Compatibility: Existing techniques are incompatible with error handling and reporting features.

Our Approach

Length-limiting Randomization: Limit the utility of any disclosed address.

Limit Disclosures in EH-metadata: Intra-block randomization, reduce EH-metadata stored in memory.

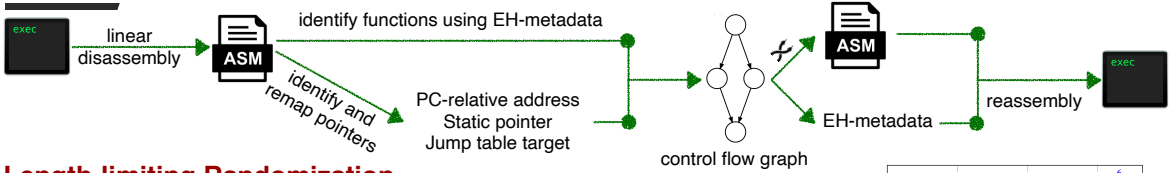
New Entropy Metrics: To quantify security against the new threat model EH-metadata leakage.

Binary Analysis and Instrumentation: Compatible with x86-64 binaries with error handling and reporting.

Key Benefits

Compatibility with COTS binaries, including low-level libraries with hand-written assembly.
Compatibility with exceptions and stack traces.
Strong Security against EH-metadata leakage.
Low Runtime Overhead (less than 5%)

Binary Analysis and Instrumentation



Length-limiting Randomization

Bounded utility of disclosed address: Break every function into partitions of k instructions on average.

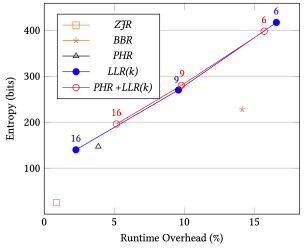
Tunable entropy and performance: Tune k for trade-offs between security and performance.

Higher entropy for the same number of partitions: Additional randomness in the placement of breaks.

Can be combined with other randomizations: LLR introduces enough breaks for same partition size.

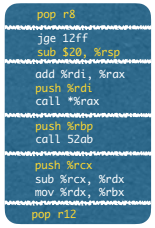
larger *k*
faster, leak *more*

smaller *k*
slower, leak *less*



SPECspeed 2017

Limiting Disclosures in EH-metadata



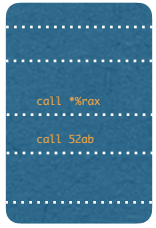
Available in 95% of Linux system binaries
12 blocks per function on average

Reducing EH-metadata

Exception throw is done via a call instruction
Store EH-metadata only for call-containing blocks
Expand call-containing blocks to adjacent blocks
Restore full EH-metadata to support stack tracing

Support exception handling

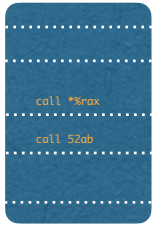
Readable memory



Original EH-metadata

Non-readable memory

Same time, space overhead
Leaks not reveal randomization



Replace to support stack tracing