Practical Fine-Grained Binary Code Randomization

Soumyakant Priyadarshan, Huan Nguyen, R. Sekar
SecLab

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**

- Linear disassembly
- Identify functions using EH-metadata
- Identify and remap pointers
- PC-relative address
- Static pointer
- Jump table target
- Control flow graph
- EH-metadata reassembly

**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.

**Limiting Disclosures in EH-metadata**

- **Readable memory**
- **Non-readable memory**
- **Support exception handling**
- **Same time, space overhead**
- **Leaks not reveal randomization**

---

**Specspeed 2017**