Practical Byte-Granular Memory Blacklisting using Califorms

Hiroshi Sasaki, Miguel A. Arroyo, Mohamed Tarek Ibn Ziad, Koustubha Bhat, Kanad Sinha, Sinha Sethumadhavan
Columbia University

THE PROBLEM

- Memory safety vulnerabilities are very easy for programmers to introduce unknowingly
- A need for a lower overhead and finer-grained (i.e. intra-object) level of memory safety

OUR CONTRIBUTIONS

- Califorms offers memory safety by detecting accesses to dead bytes in memory. Blacklisted locations are not stored beyond the L1 data cache and are identified using a special header in the L2 cache (and beyond) resulting in very low overhead.

- The conversion between these formats happens when lines are filled or spilled between the L1 and L2 caches. The absence of blacklisted locations results in the cache lines stored in the same natural format across the memory system.

RESULTS

- Provides intra-object (i.e. field level) memory safety at low overheads (~1.02x-1.16x)
- Califorms is agnostic of architecture width and can be deployed over a diverse device environment

SECURITY BENEFITS

- Califorms have applications other than memory safety
  ○ Information Flow Tracking

HARDWARE

L1⇒L2 Conversion Algorithm
1. Read the cache metadata for the inserted line and set all bytes to zero.
2. If result is 0 bytes:
   3. Set the line as clean and set caliform bit to zero
   4. else:
   5. Set caliform bit to one
   6. If non-security bytes (N) < 4 bytes:
      7. Generate metadata for N non-security bytes
      8. Store metadata of first N bytes in locations obtained in 7
      9. Fill the first N bytes based on Figure 7
   10. Fill the first four bytes based on Figure 7 (use the sentinel) to mark the remaining security bytes
   11. end

L2⇒L1 Conversion Algorithm
1. Read the Caliform bit for the inserted line
2. If result is 0 bytes:
3. Set the Caliform metadata bit vector to [0]
4. else:
5. Check the least significant 2-bit of byte 0
6. Get the metadata of byte[byteAddr[0]-2] to one based on 5
7. Set the metadata of byte[byteAddr[0]] to zero
8. Set the data of byte[byteAddr[0]] to byte[byteAddr[0]-2]
9. Set the new locations of byte[byteAddr[0]] to zero
10. end

SOFTWARE

- On allocation, a special instruction de-blacklists bytes.
- To maintain compatibility, we (de)serialize data passed externally.
- On deallocation, we revert bytes as blacklisted.

IMPACT

- Scalable
  ○ Architecture width agnostic
  ○ Low overheads
- Califorms have applications other than memory safety
  ○ Information Flow Tracking