VIP: Safeguard Value Invariant Property for Thwarting Critical Memory Corruption Attacks

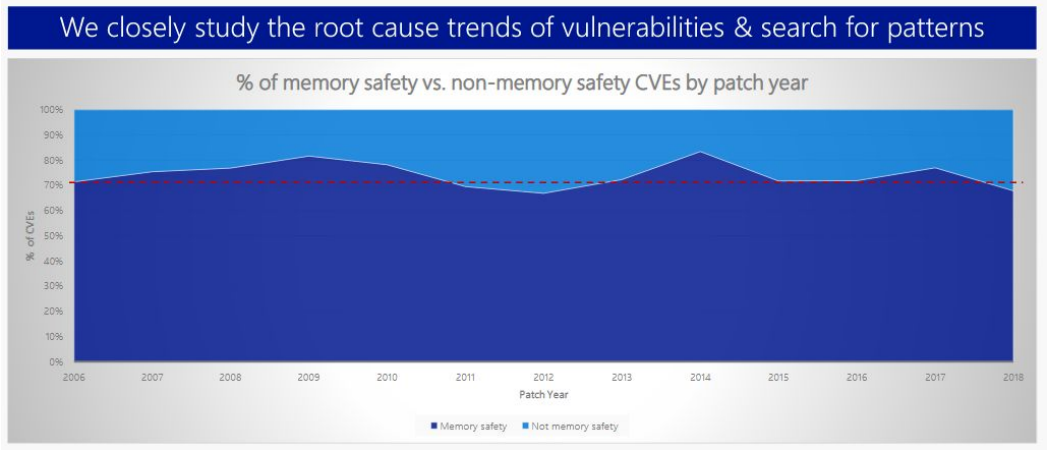
Mohannad Ismail (Virginia Tech), Jinwoo Yom (Virginia Tech), Christopher Jelesnianski (Virginia Tech), Yeongjin Jang (Oregon State University), Changwoo Min (Virginia Tech)

CCS '21: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security

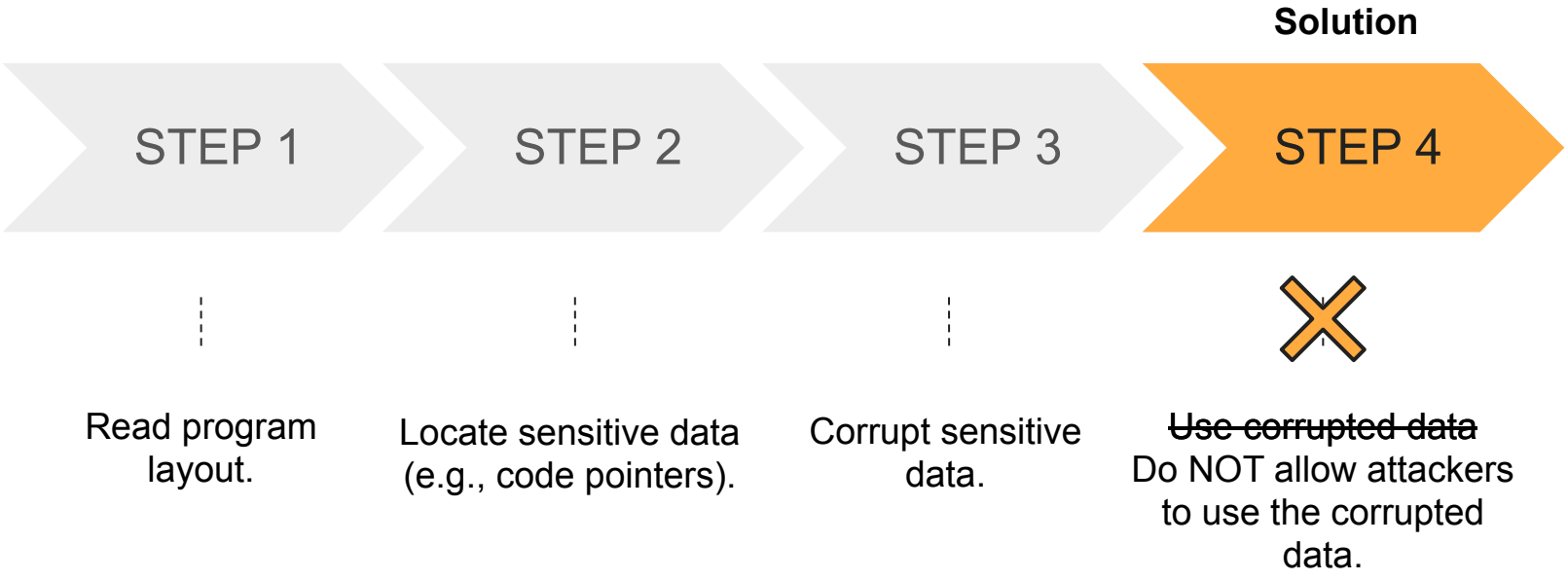
Memory corruption vulnerability is root of all EVIL

- Microsoft reported that **70%** of all security bugs are due to various **memory safety** issues.

- Top three memory corruption attacks:
  - Heap out-of-bounds.
  - Use-after-free.
  - Type confusion.

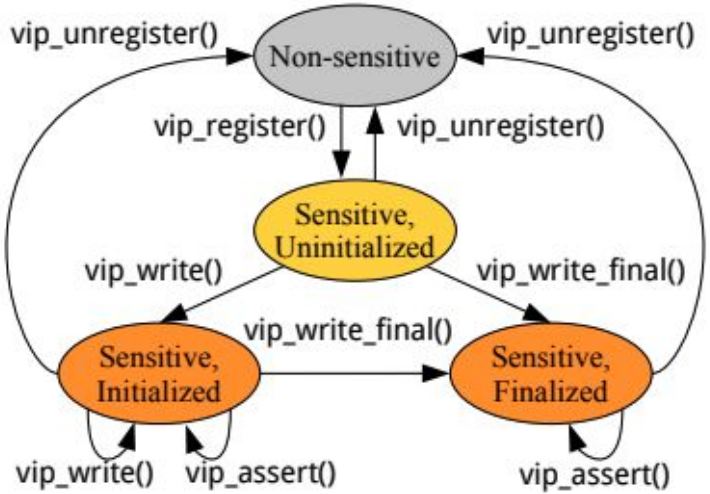


Breaking an essential step in memory corruption attacks



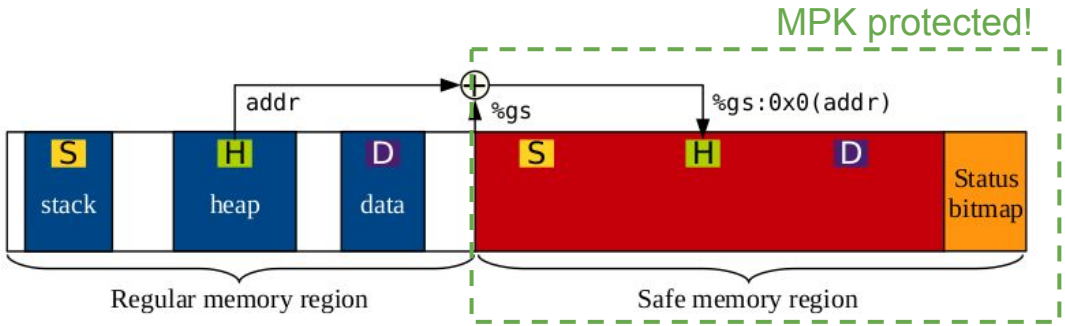
Value Invariant Property (VIP)

- Our intuition behind VIP originates from a common pattern in programs:
  - *Security-sensitive data should never be changed between two legitimate writes so there is a period such that security-sensitive data is immutable.*
- This period is represented by the state transition diagram, that relies on VIP primitives to enforce value integrity of security-sensitive data.



Overview of Value Invariant Property (VIP)

- VIP maintains a shadow copy of sensitive data in an isolated “safe” region.
- VIP checks and verifies value integrity instead of tracking control-flow.
- The safe region is protected with Intel MPK, so we can unlock the safe region when access is needed by the program, and then lock it to prevent illegitimate access.



HyperSpace

- Fully implemented prototype that enforces VIP.
- Four Defense Mechanisms:
  - Control flow integrity (VIP-CFI)
    - Protects ***all*** code pointers.
  - C++ VTable pointers protection (VIP-VTPtr)
    - Protects ***all*** virtual function table pointers (VTPtrs).
  - Code pointer integrity (VIP-CPI)
    - VIP-CFI and VIP-VTPtr protection + ***all*** sensitive object pointer protections.
  - HyperSpace heap metadata protection
    - Protect against inline heap metadata corruption attacks.

Optimizations and evaluation

- Frequent usage of `wrpkru` to modify safe region can incur significant overhead.
- **Six major optimizations:**
  - Inlining DVI functions.
  - Not instrumenting objects in the SafeStack.
  - Runtime checks to reduce permission changes.
  - Coalescing permission changes within a Basic Block.
  - Coalescing permission changes within a safe function.
  - Huge Page enabled.

Reduces `wrpkru` usage!

Security evaluation with:
- 3 real-world exploits (CVEs)
- 6 synthesized attacks

Performance/Memory evaluation with:
- SPEC CPU 2006
- NGINX (v1.14.2)
- PostgreSQL

Conclusion

- Value Invariant Property (VIP) is a new defense policy that provides a versatile and elegant solution to thwarting memory corruption exploits.
- Our prototype, HyperSpace, enforces VIP to provide various security mechanisms with the strongest guarantee (VIP-CPI) with low performance and memory overhead.
- Contributions:
  - VIP
  - HyperSpace
  - Optimization of HyperSpace
  - Thorough evaluation of HyperSpace

Thank You !