ENCORE: ENhanced program protection through COmpiler-REwriter cooperation

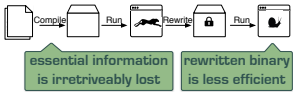
PIs: **Michael Franz** (University of California, Irvine), **Kevin Hamlen** (University of Texas at Dallas), **Mathias Payer** (Purdue University)

Problem

- Consumers of software, for which no source code is available must wait for vendors to fix vulnerable programs.
- In 2014, the top 5 **zero-day vulnerabilities took 59 days to patch** on average.
- Total window of vulnerability = 295 days.
- Binary rewriting could **close the window of vulnerability** if the techniques were practical.

Current Practice

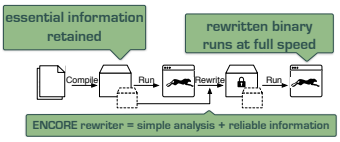
- **Structural information is discarded** by compiler: speed and size is all that matters.
- Binary rewriters try to recover structural information using **complex analysis plus unreliable guesses.**
- Rewritten programs take up more space and run slower than their original counterparts.



Proposed Practice

Change the way compilers have been constructed the last sixty-odd years:

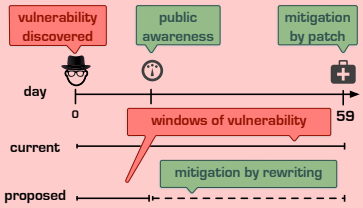
1. Compiler **retains structural information.**
2. Simple binary analysis identifies **what info can be reliably recovered** and what cannot.
3. Residual "hard to recover" **information embedded in output** ENCORE binary.



Applications

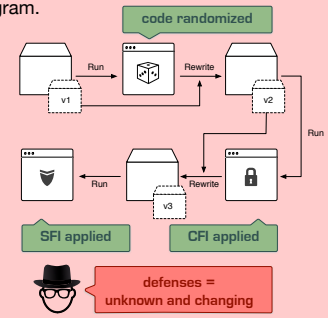
Shorten window of vulnerability

In the time window from the discovery of a vulnerability to the availability of a patch, we can apply general-but-costly mitigations such as bounds checking while a patch is developed, tested, and distributed.



Moving Target Defense

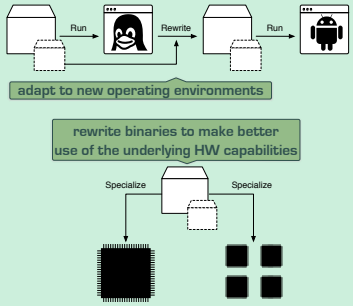
Challenge: adversaries know what mitigations they will need to bypass to exploit a particular program.



Adaptation and Optimization

Challenges:

- Software often outlives hardware.
- **Applications break** when host system changes.
- Software **compiled for lowest common denominator** makes poor use of actual hardware.

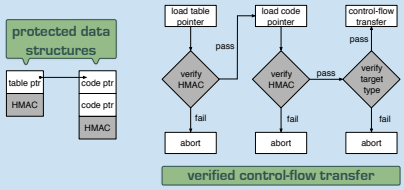


Intermediate Results

Protecting Control Flow

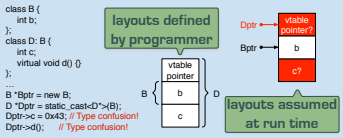
Enforce **forward-edge control-flow integrity**:

- Embed list of permissible control-flow edges.
- Protect integrity of data structures containing code pointers.
- Verify indirect control-flow transfers for C++ programs [1], C systems software (OS/VMM) [2], Objective-C programs [3].



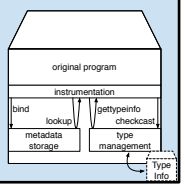
Enforcing Type Safety

Type confusion bugs are emerging as an important attack vector for C++ programs.



Challenge: correctness of many downcasts is difficult to check dynamically.

Solution: embed type info to allow checking of downcasts at run time.



Risks and rewards

ENCORE binaries:

- ✓ Close or **shorten window of vulnerability.**
- ✓ Present adversaries with **moving target.**
- ✓ **Facilitate adaptation** of legacy binaries.
- ✓ Enable binary rewriting without developer assistance.

The additional information in ENCORE binaries may facilitate reverse engineering. However, a tunable range of disclosure options and obfuscation allow intellectual property concerns to be traded-off against performance.

Publications

1. C. Zhang, S. Carr, T. Li, Y. Ding, C. Song, **M. Payer**, and D. Song. "*VTrust: Regaining trust on virtual calls.*" In *NDSS* 2016.
2. X. Ge, N. Talele, **M. Payer**, and T. Jaeger. "*Fine-Grained Control-Flow Integrity for Kernel Software.*" In *EuroS&P* 2016.
3. J. Lettner, B. Kollenda, A. Homescu, P. Larsen, F. Schuster, L. Davi, A.-R. Sadeghi, T. Holz, and **M. Franz**. "*Subversive-C: Abusing and Protecting Dynamic Message Dispatch.*" In *USENIX ATC* 2016.
4. S. Volckaert, B. Coppens, A. Voulimeneas, A. Homescu, P. Larsen, B. De Sutter, and **M. Franz.** "*Secure and Efficient Application Monitoring and Replication.*" In *USENIX ATC* 2016.
5. N. Carlini, A. Barresi, **M. Payer**, D. Wagner, and T. Gross. "*Control-flow bending: On the effectiveness of control-flow integrity.*" In *USENIX SEC* 2015.
6. I. Haller, Y. Jeon, H. Peng, **M. Payer**, C. Giuffrida, H. Bos, and E. van der Kouwe. "*TypeSan: Practical Type Confusion Detection.*" In *ACM CCS* 2016.
7. K. Braden, S. Crane, L. Davi, **M. Franz**, P. Larsen, C. Liebchen, and A.-R. Sadeghi. "*Leakage-Resilient Layout Randomization for Mobile Devices.*" In *NDSS* 2016.

Interested in meeting the PIs? Attach post-it note below!

NSF Secure and Trustworthy Cyberspace Principal Investigators' Meeting
Jan. 9 -11th 2017
Arlington, VA

National Science Foundation
WHERE DISCOVERIES BEGIN

UT DALLAS
PURDUE UNIVERSITY