Stony Brook University

HEXLAB

Principled and Practical Software Shielding against Advanced Exploits

Michalis Polychronakis
Stony Brook University

Cybersecurity TTP Acceleration Workshop – 17 April 2018

Motivation

Software vulnerability exploitation

- Among the leading causes of system compromise and malware infection

We have to live with C/C++

- Performance, compatibility, developer familiarity, vast existing code base, …
- Many memory-safe programming languages exist, but full transition would require an immense rewriting effort
- Unlikely to happen for core systems code, resource-constrained IoT devices, …

Defending against Vulnerability Exploitation

Finding and killing bugs

- Bug bounties, sanitizers, fuzzing, symbolic exec, …
- *Who* will find the next 0-day?

Retrofit memory safety to C/C++

- Eradicate the root cause of the problem: *memory errors*
- Performance and compatibility challenges, but promising steps are being made

Exploit mitigations

- Assuming a vulnerability exists, "raise the bar" for exploitation
- DEP, GS, SafeSEH, SEHOP, ASLR, CFI, …

Exploit Mitigations Do Raise the Bar…

Pwn2Own 2007

> *"A New York-based security researcher [Dino Dai Zovi] spent less than 12 hours to identify and exploit a zero-day vulnerability in Apple's Safari browser"* [1]

Exploit Mitigations Do Raise the Bar…

Pwn2Own 2007

"A New York-based security researcher [Dino Dai Zovi] spent less than 12 hours to identify and exploit a zero-day vulnerability in Apple's Safari browser" [1]

Pwn2Own a decade later

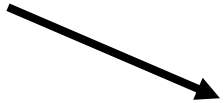
"This year saw several teams sponsored by their employers participating" [2]

[1] https://www.theregister.co.uk/2007/04/20/pwn-2-own_winner/
[2] https://blog.trendmicro.com/pwn2own-2017-event-ages/

…but Attackers Can Often Knock the Bar Off

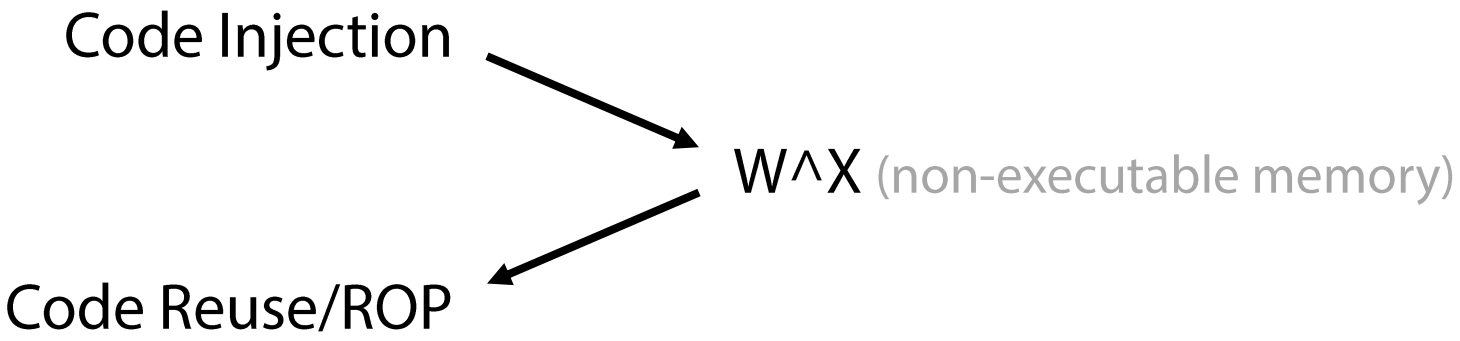
Code Injection

…but Attackers Can Often Knock the Bar Off

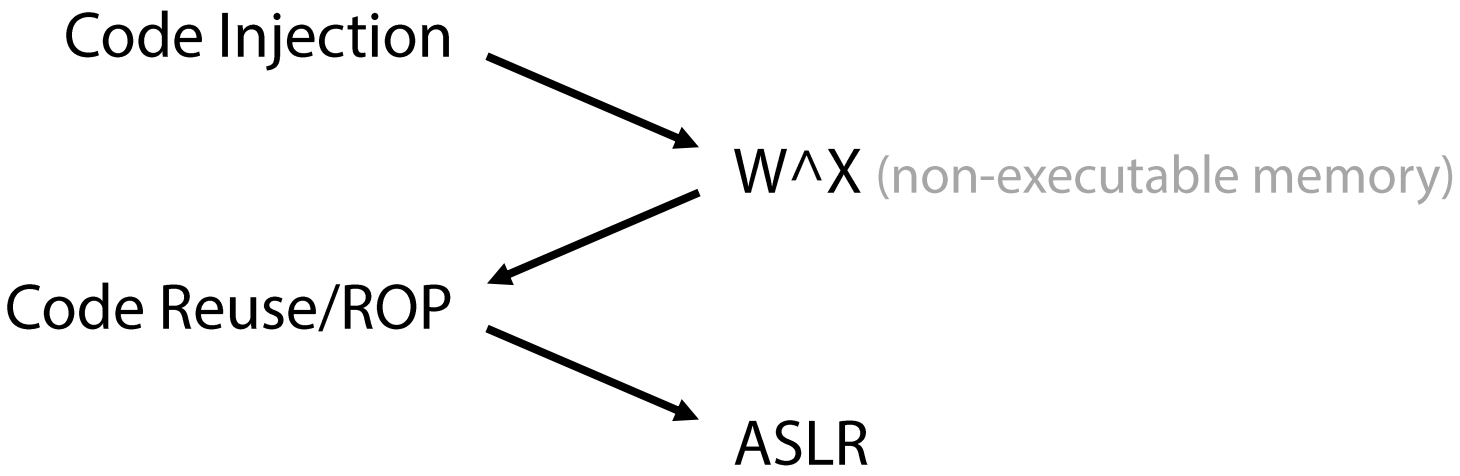
Code Injection 

W^X (non-executable memory)

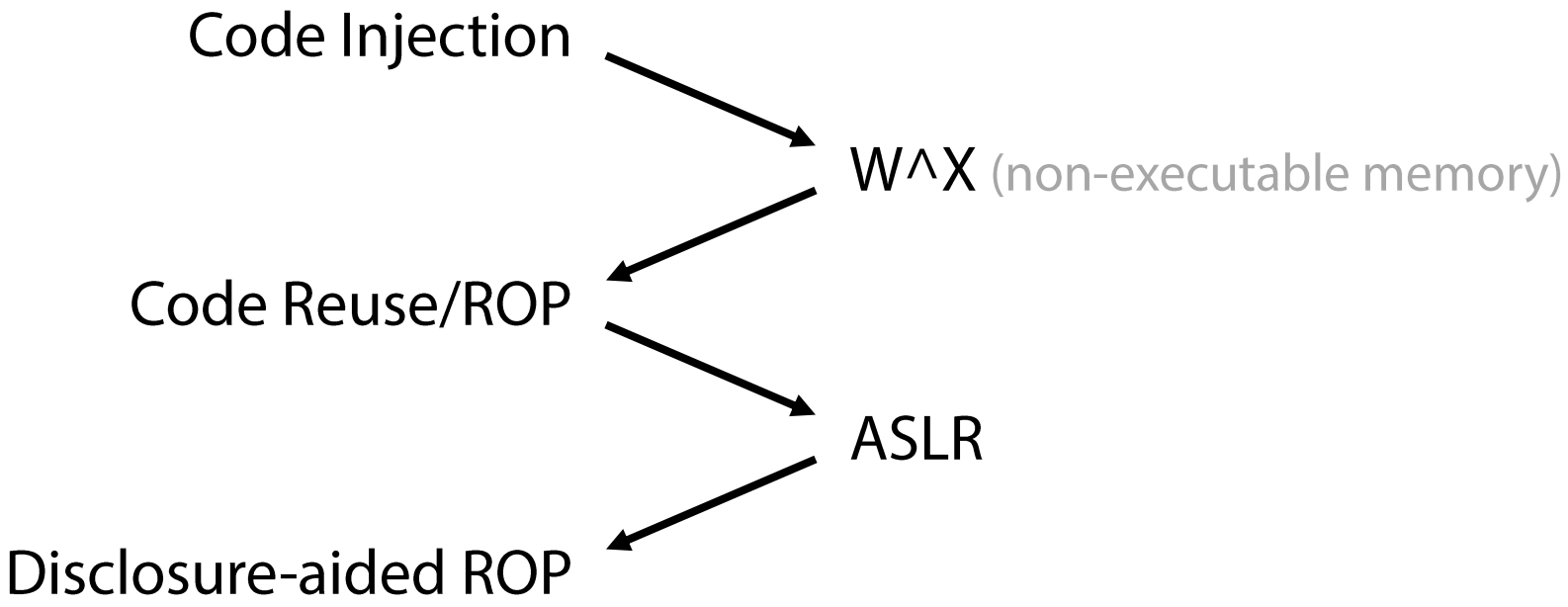
…but Attackers Can Often Knock the Bar Off



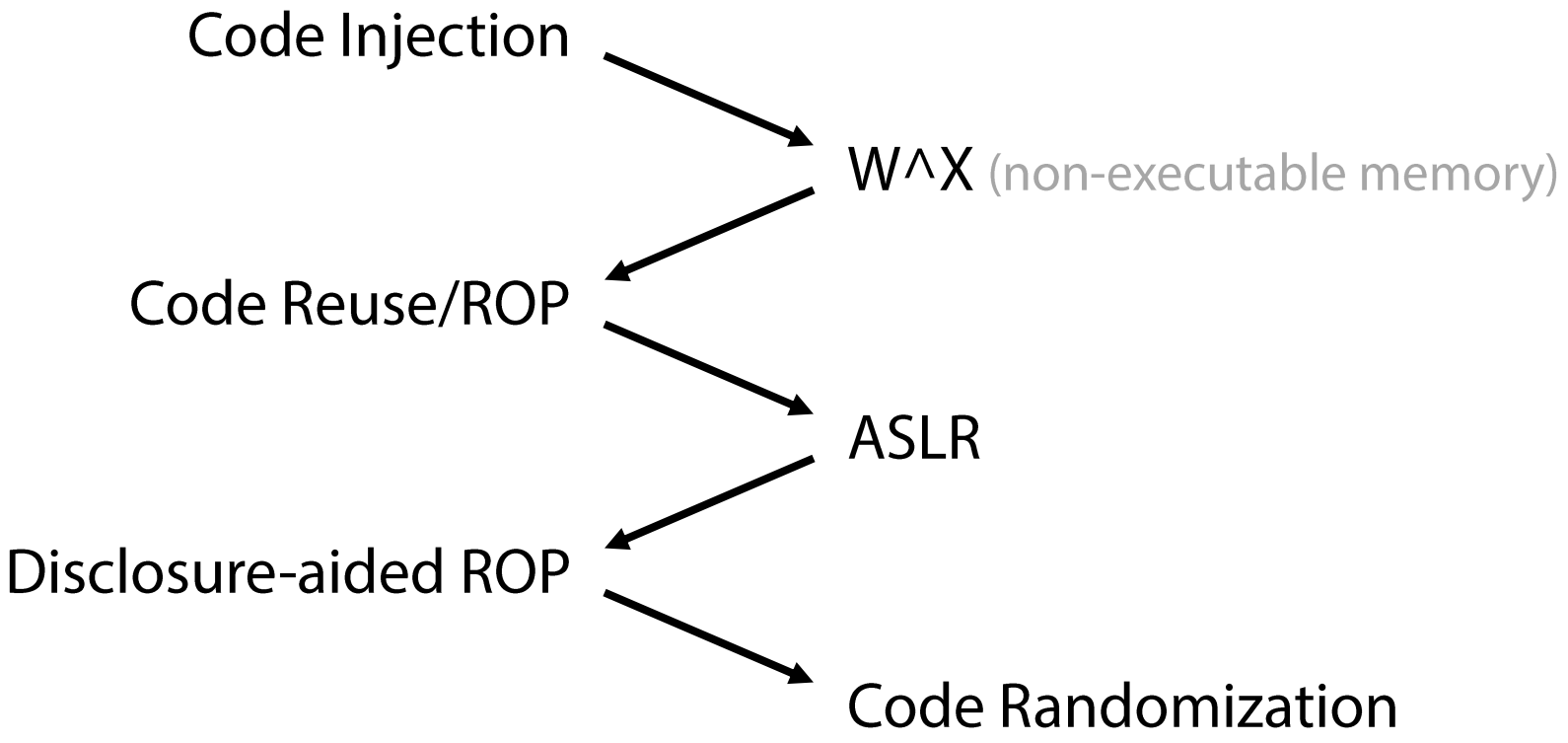
…but Attackers Can Often Knock the Bar Off



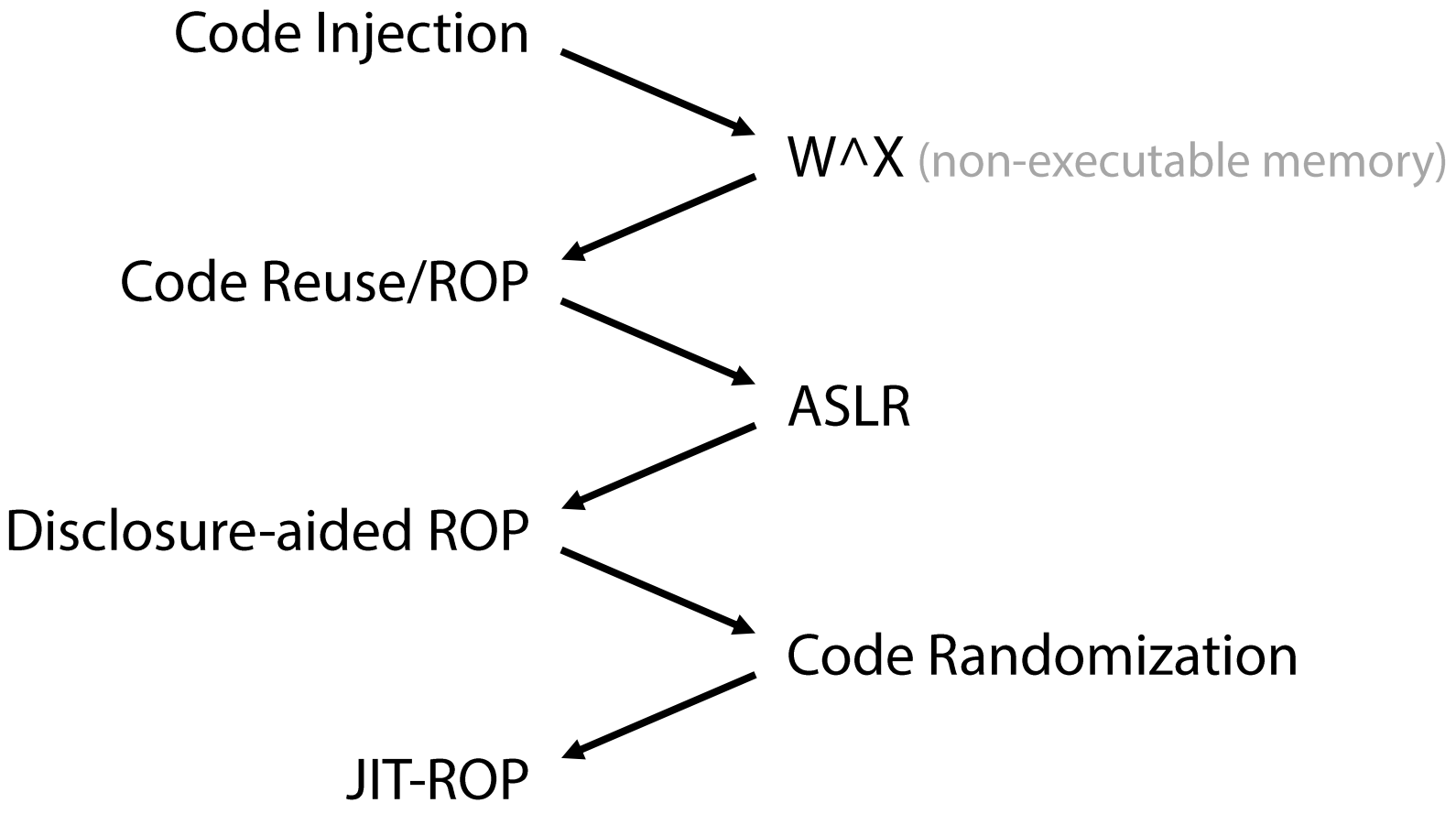
…but Attackers Can Often Knock the Bar Off



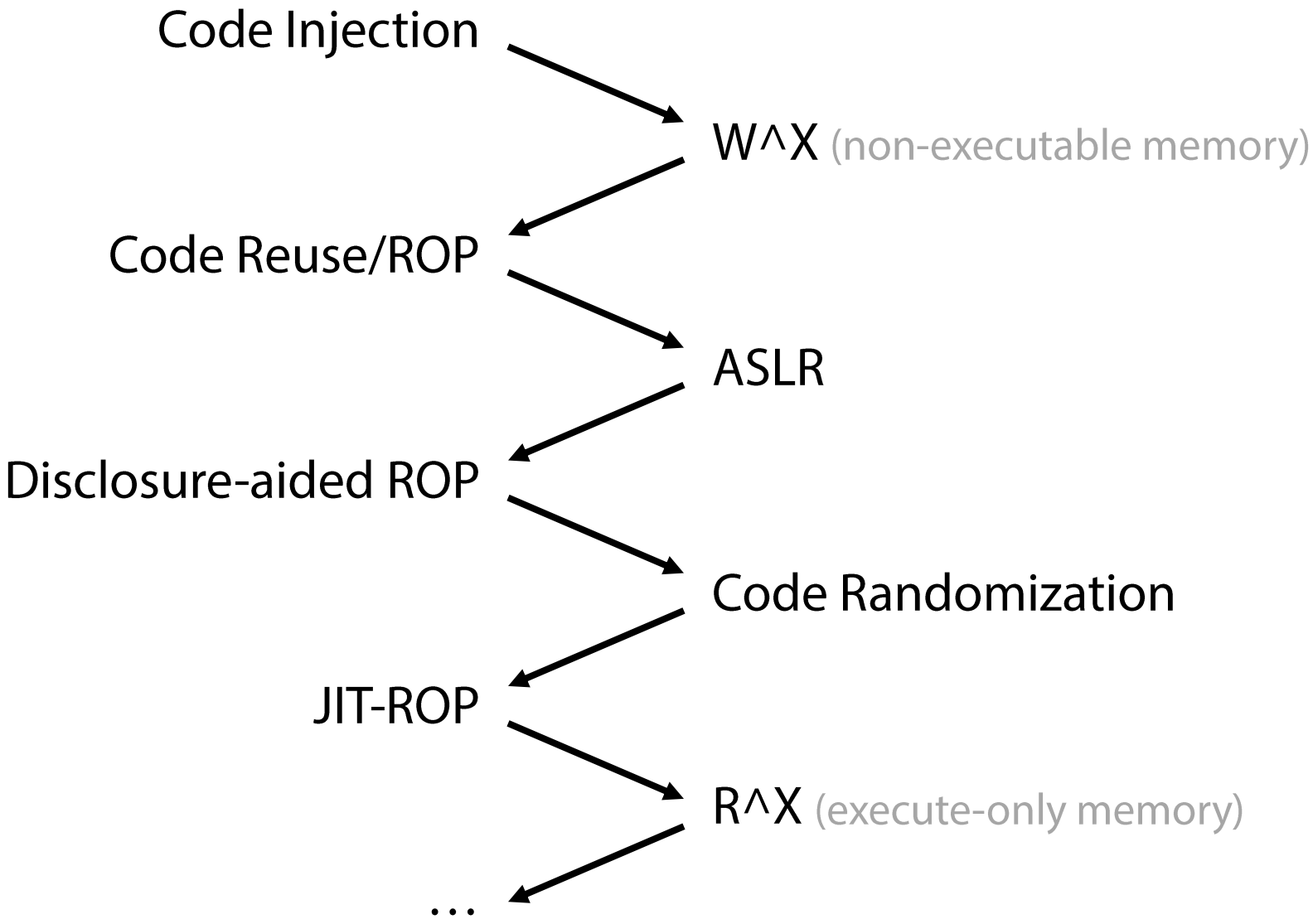
…but Attackers Can Often Knock the Bar Off



…but Attackers Can Often Knock the Bar Off



…but Attackers Can Often Knock the Bar Off



Main Research Objectives

Design novel software shielding techniques

- Code diversification: *undermine adversaries' assumptions*
- Unneeded code and logic removal: *reduce the attack surface*
- Data protection: *keep sensitive data out of reach*

Focus on emerging exploitation techniques

- Disclosure-aided exploitation
- Data-only attacks

Enable their practical applicability on commodity software and systems

- Alleviate current deployment obstacles faced by protections that break software uniformity

Use Case: Code Diversification

Function/basic block/instruction reordering, instruction substitution, register reassignment, …

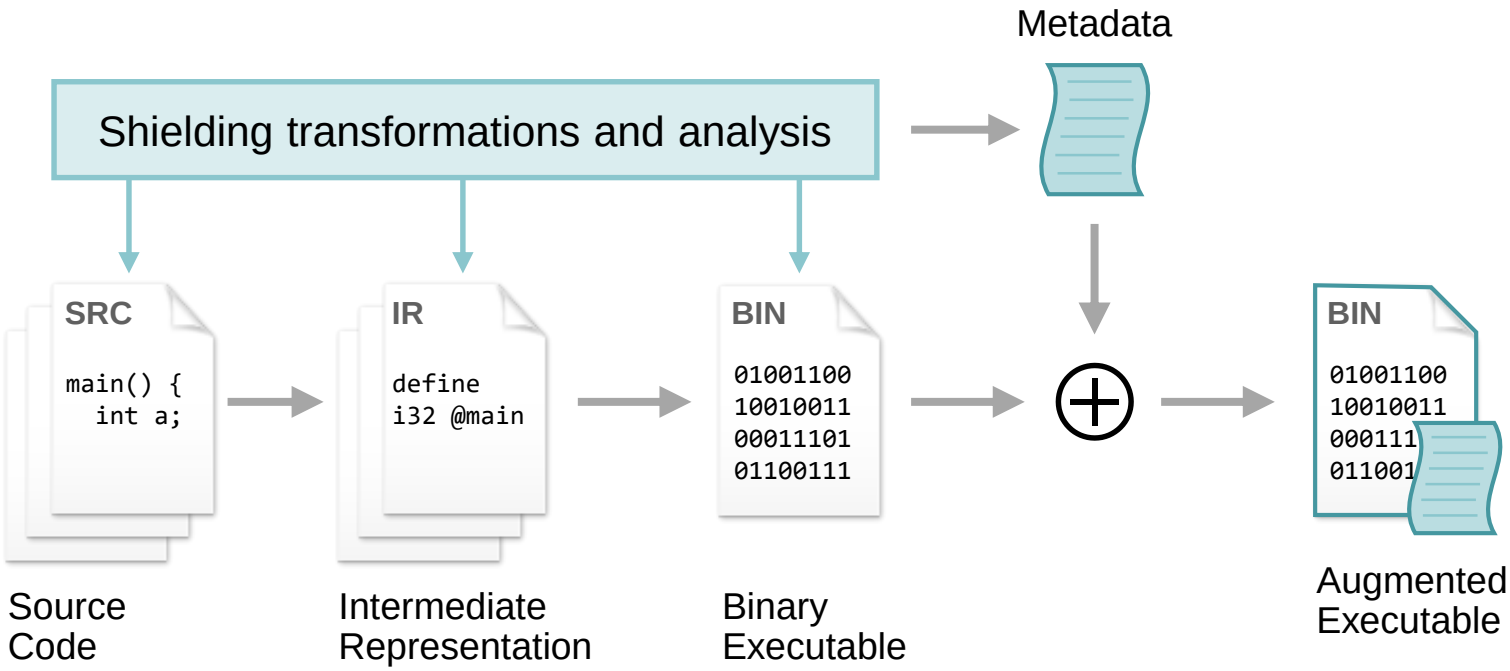
Effective mitigation against code reuse attacks

- JIT-ROP can circumvent it
- *Prerequisite* of execute-only memory protections for defending against JIT-ROP

Despite decades of research, still not deployed

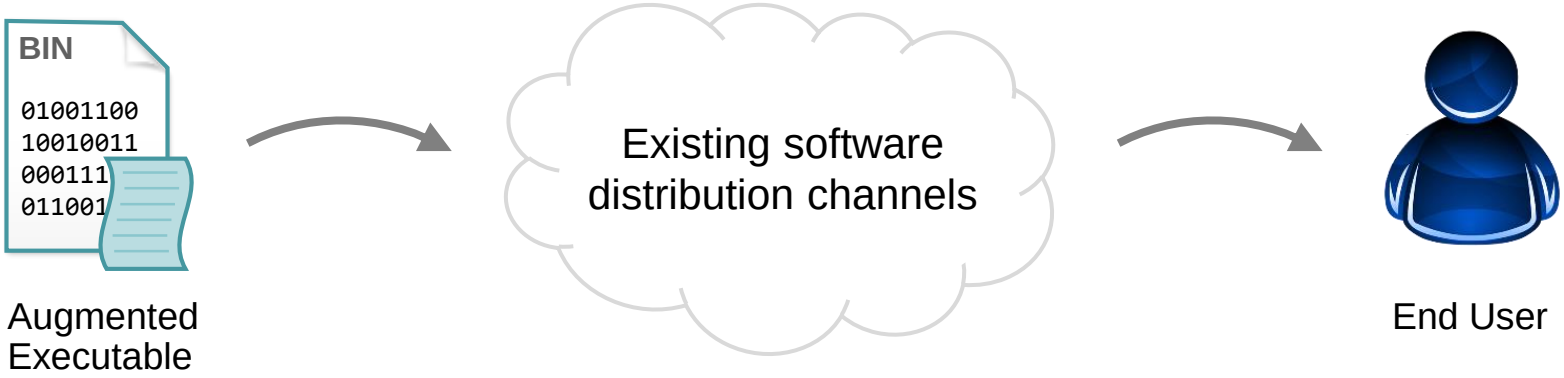
- Lack of a *transparent* deployment model ➔ users are responsible for diversifying their software
- Incompatible with debugging, crash reporting, whitelisting, and other mechanisms that rely on software uniformity

Metadata-assisted Binary Transformation (1/3)



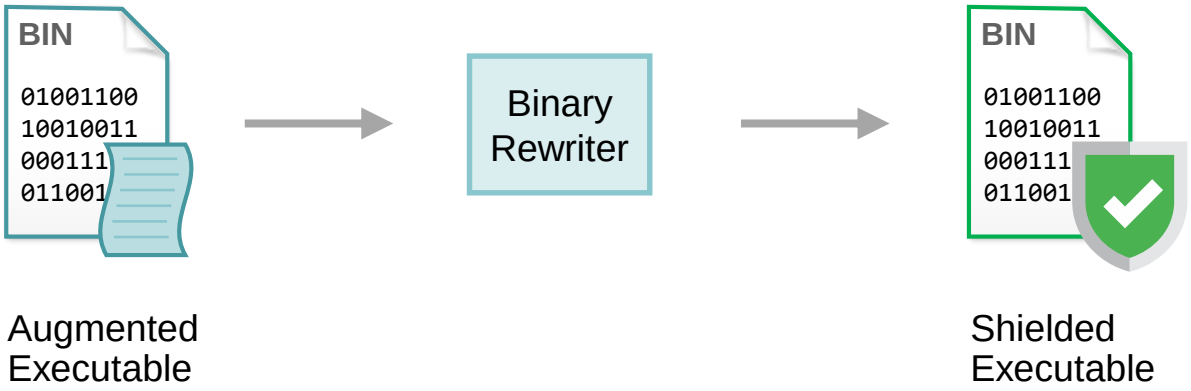
Software vendors still release a single "master" executable, augmented with transformation-assisting metadata

Metadata-assisted Binary Transformation (2/3)



Augmented executables are delivered in the same way as before, through the same software distribution channels, while patching/updating is not affected

Metadata-assisted Binary Transformation (3/3)



At the client side, a binary rewriter leverages the embedded metadata to rapidly generate a specialized (debloated/randomized/shielded) executable

Summary

Design novel software shielding techniques, and enable their practical applicability to commodity software and systems

Upcoming IEEE S&P '18 paper and code release

> *Compiler-assisted Code Randomization.* Hyungjoon Koo, Yaohui Chen, Long Lu, Vasileios P. Kemerlis, Michalis Polychronakis
>
> `https://github.com/kevinkoo001/CCR`

CAREER: Principled and Practical Software Shielding against Advanced Exploits
NSF CNS-1749895, $499,899 (6/1/2018 – 5/31/2023).

HEXLAB