CMPT 479 - ROPHunter

Kattie Sepehri; Leo Goldberg; Samantha Yu

Motivation

- W ⊕ X is a widely used defence against traditional code injection attacks
- ROP is an extension to traditional return-to-libc attacks
- **Goal:** Reproduce the ROP gadget finder from [Shacham, 2007]
  - Requires *no function calls* to perform a return-to-libc attack and can easily bypass W ⊕ X
- **Importance:** libC exists on all Linux-based machines

Problem

Objective: Find a set of ROP gadgets in libC which can be chained together to perform operations

Input: Binary for libC
Requirement for the attack: Buffer overrun
Output: Chained ROP Gadgets that perform an operation (eg. opening a shell)

Challenges

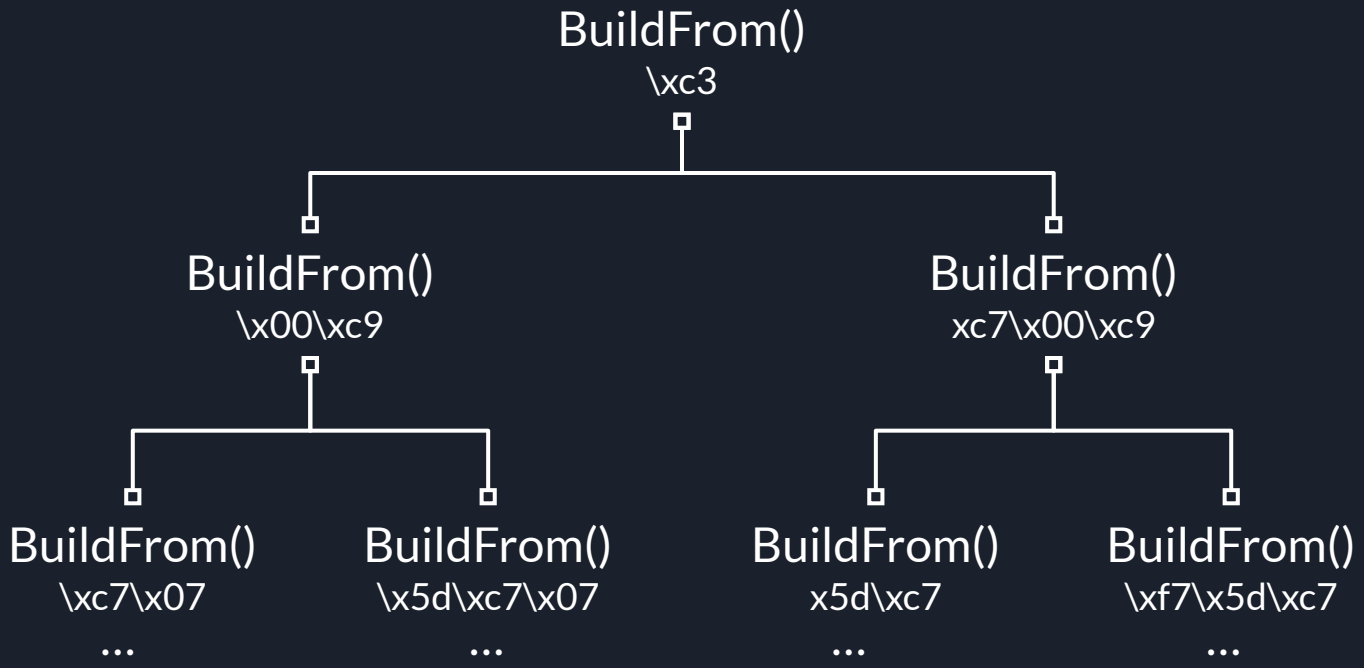
- Determining the addresses of the gadgets found
- Maintaining a low runtime relative to existing ROP gadget finders
- Chaining gadgets together to perform operations will increase run time (to be added)
- Testing the chained gadgets in a buffer overrun attack

Solution

- Galileo algorithm from [Shacham, 2007]:
  - For each ret (C3) instruction:
    - Call BuildFrom()
- BuildFrom():
  - Traverse through each byte backwards starting from ret
    - If byte sequence is a valid and non-boring instruction:
      - Insert instruction into trie
      - Call BuildFrom()

Solution - Call Graph/Trie

```
code = \xf7\x5d\xc7\x07\x00\xc9\xc3
```



Evaluation Setup

	[Shacham, 2007]	ROPgadget	ROPHunter
libc	Fedora Core 4's libc-2.3.5.so	Ubuntu 18.04's libc-2.27.so	Ubuntu 18.04's libc-2.27.so
libc Executable Segment Size	1,189,501 bytes	N/A	1,542,508 bytes
Disassembler	libdisasm	Capstone	Capstone
Trie	N/A	Own library	Pygtrie
CPU	1.33 GHz PowerPC G4 with 1 GB RAM	2.20 GHz Intel i7 with 16.0 GB RAM	2.20 GHz Intel i7 with 16.0 GB RAM

Preliminary Results

	[Shacham, 2007]	ROPgadget	ROPHunter
Time	1.6s	8.246s	32.579s
Gadgets Found	15,121	116,901	185,860
Time Per Gadgets	105.81 µs/Gadget	70.538 µs/Gadget	175.287 µs/Gadget
False Positives/ Negatives	N/A	TBD	TBD

High Running Time

- Cause: Path explosion
  - Solution: Limiting number of instructions per gadget
- Possible Causes:
  - Pygtrie vs. ROPgadget's own trie library
  - Capstone's disasm() vs. disasm_lite()

Extra Gadgets Returned

- Possible Causes:
  - Size of libc's executable segment
  - ROPgadget optimizations

Questions?