RETURN-ORIENTED PROGRAMMING (ROP) OVERVIEW

Agenda

- Last cycle:
  - Discussed basic stack buffer overflow
- This cycle:
  - Mitigations that prevent those older attack techniques
  - Discuss ROP and why it's used
  - Tool Demonstration

Stack Buffer Overflows

- Last cycle we demonstrated basic attack
- Common mitigations now prevent that attack:
  - Data Execution Prevention (DEP)
  - Write XOR Execute (W^X)
  - NX bit, XD bit, XN
- Can’t write shellcode to stack AND execute that shellcode

Return-Oriented Programming

- Write addresses/data to stack
- Execute instructions that already exist on the system

- ROP gadgets are the key!
  - Sequence of machine instructions that end in a return instruction

Example

STACK

0x33333330	0xB8DEADBE
0x33333334	0xEF909090
0x33333338	SAVED_EBP
0x3333333C	0x30333333
0x33333440	… stuff …
0x33333444	… stuff …

- Goal:
  Write “0xDEADBEEF” into EAX
- Shellcode:
  B8DEADBEEF mov eax, 0xEFBEADDE
- But if we WRITE that shellcode instruction to the stack, then we cannot EXECUTE that instruction as well

Example

- GOAL:
  Write “0xDEADBEEF” into EAX

- ROP:
  WRITE: (during overflow)
  “DEADBEEF” to the stack
  EXECUTE: (after return)
  0x58 pop eax
  0xC3 ret

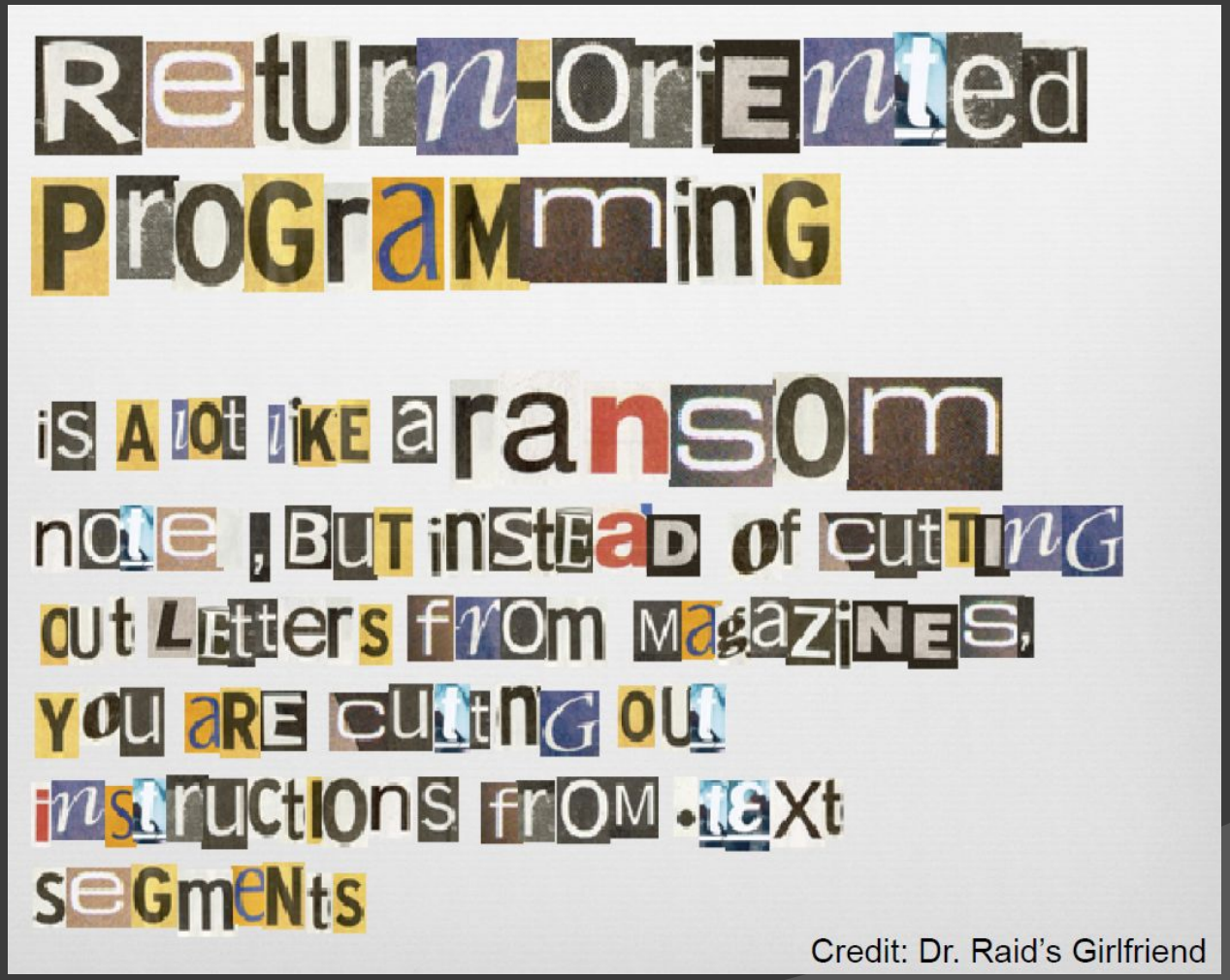
STACK

0x33333330	0x90909090
0x33333334	0x90909090
0x333333380	SAVED_EBP
0x3333333C	0x88888888
0x33333440	0xDEADBEEF
0x33333444	0x94888888

ADDRESS

0x88888884		… stuff …
0x88888888	58	pop eax
0x8888888C	C3	ret
0x88888890		… stuff …
0x88888894	CC	breakpt

ROP Analogy



Cite: Dai Zoni, D. (2010). Return-oriented Exploitation. *Blackhat Conference.* Retrieved from https://media.blackhat.com/bh-us-10/presentations/Zovi/BlackHat-USA-2010-DaiZovi-Return-Oriented-Exploitation-slides.pdf

ROP Made Possible By…

- Data can be interpreted as instructions
  - 0xC3
    - Instruction: RET
    - Integer: 195
    - Character: “Ã”

ROP Made Possible By…

- Variable-length instructions introduce unintended instructions
  - B8 89 41 08 C3
    - mov eax, 0xC3084189
  - 89 41 08 C3
    - mov [ecx+8], eax
    - ret

Cite: Dai Zoni, D. (2010). Return-oriented Exploitation. *Blackhat Conference.* Retrieved from https://media.blackhat.com/bh-us-10/presentations/Zovi/BlackHat-USA-2010-DaiZovi-Return-Oriented-Exploitation-slides.pdf

ROP Bottom-line

Preventing the introduction of malicious code

IS NOT SUFFICIENT

to prevent the introduction of malicious computation.

Cite: Shacham, H. (2007, October). The geometry of innocent flesh on the bone: Return-into-libc without function calls (on the x86). In *Proceedings of the 14th ACM conference on Computer and communications security* (pp. 552-561). ACM. Retrieved from https://cseweb.ucsd.edu/~hovav/dist/geometry.pdf

Tool Demonstration

Tool Demonstration - Compile

```
gcc -g -fno-stack-protector -z
execstack mybigecho.c -o mybigecho

gcc -g -fno-stack-protector
mybigecho.c -o mybigechoNX
```

Tool Demonstration - NX mitigation

```
gdb ./mybigecho
   run < payload
   //notice “Howdy Texas!” printed
   quit
gdb ./mybigechoNX
   run < payload
   //notice SEGFAULT
   quit
```

Tool Demonstration - ROPGadget

- sudo pip install capstone
- sudo pip install ropgadget
- ROPgadget --binary myecho
- ROPgadget --binary myecho --only “pop|ret”

```
perl -e ' print "\x90"x264; print
"\xd8\xf3\xff\xbf"; print
"\xA1\x84\x04\x08"; print
"\xEF\xBE\xAD\xDE"; print
"\x9f\x84\x04\x08"; ' > payload_ROP
```

Tool Demonstration - NX mitigation

```
gdb ./mybigechoNX
   break mybigecho
   run < payload_ROP
   info reg    //notice EAX=1
   cont
   //notice SIGTRAP
   info reg    //notice EAX=0xDEADBEEF
   quit
```

Summary

3 Main Ideas

- DEP and W^X mitigations prevent traditional shellcode stack-based buffer overflow attacks
- Return-oriented Programming can bypass DEP and W^X mitigations
- Tools can help enable ROP

Future Work

- Discuss how Address Space Layout Randomization (ASLR) and Stack Canaries affect the scenario
- Explore Jump-oriented Programming
- Demonstrate additional ROP tools, such as ROPEME, mona.py, and ropper
- Mitigations that make ROP more difficult, such as the Enhanced Mitigation Experience Toolkit (EMET)

References

- Dai Zoni, D. (2010). Return-oriented Exploitation. *Blackhat Conference.* Retrieved from https://media.blackhat.com/bh-us-10/presentations/Zovi/BlackHat-USA-2010-DaiZovi-Return-Oriented-Exploitation-slides.pdf

- Shacham, H. (2007, October). The geometry of innocent flesh on the bone: Return-into-libc without function calls (on the x86). In *Proceedings of the 14th ACM conference on Computer and communications security* (pp. 552-561). ACM. Retrieved from https://cseweb.ucsd.edu/~hovav/dist/geometry.pdf

Post Questions and Comments to the Discussion Board