

Lecture 08 Code Reuse (part 1)

CTF crunch	Computer and Network Security November 18, 2019 Computer Science and Engineering Department	
CSE Dep. ACS, UPB	Lecture OS, Code Resee (pert 1) 1/23	
CNSO	Defense Mechanisms	Notes
static & dynamiASCII armored a	address space	
 stack guard, can DEP: Data Exec 		
ASLN. Address	Space Layout Kandomization	
CSE Dep. ACS, UPB	Lecture Oli, Code Resse (port 1) 2/23	
CNS Ò	Data Execution Prevention	Notes
writable code mstack, heap, dat	a, bsss	
▶ you cannot (easi	ily) inject code (and run it)	
CSE Dep, ACS, UPB	Lecture OII, Code Roose (part 1) 3/23	
CNS	Code Reuse	Notes
use existing code		
text and libraryinteresting to us	se/call functions	
smaller pieces mno need to injec		

Notes

CSE Dep, ACS, UPB Lecture 08, Code Reuse (part 1) 4/2

CNS.	return-to-libc	Notes
S 11 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		
call existing library functionscanonical exploit is calling system("/bin/sh") in	libc	
code pointer is overwriten with address of library f	unction	
CSE Dep. ACS, UPB Lecture OE, Code Reuse (part 1)	5/23	
CNS Return-Oriented	Programming	None
OTF crunch		Notes
use smaller pieces of codepieces are called ROP gadgets, ending in ret instr	uction	
▶ payload consists of data on stack and pointers to R		
CSE Day, ACS, UPB Lecture GB, Code Reset (pet 1)	6/23	
Case Long, Acts, Ortio	0/ 23	
CNS ⊕	igh-level View	
orr crunch		Notes
call functions in libraries (libc)		
call system("/bin/sh") for a shellcall puts() for information leak		
CSE Days, ACS, UPB Lecture GE, Code Please (part 1)	8/23	
	. 19	
CNS Steps in a ret-	-to-libc attack	Notes

buffer overflow required identify function addresses and addresses of arguments (such as the $^{\prime\prime}/bin/sh^{\prime\prime}$ string) ovewrite code pointer with function address ▶ place arguments on stack

<pre>irrelevant_ret, offset*"A" + p; p32(bin_sh_add: offset*"A" + p;</pre>	<pre>write_lib_address + _address + arg1 + arg2 + 32(system_address) + 4*"B" + ress) 32(write) + 4*"B" + p32(1) + s) + p32(buf_len)</pre>	
CSE Dep. ACS, UPB	Lacture 08, Code Reuse (part 1) 10/23	
CNS	Protecting Against ret-to-libc Attacks	Note:
use stack canary	omize function addresses in library leak to bypass protection mechanisms	
CSE Dup. ACS. UPB	Lecture OB, Code Reuse (part 1) 11/23 ret-to-libc Shortcomings	Note
	ciple calls due to stack limitations be too coarse; you may need smaller chunks	
CSE Dep. ACS, UPB	Lecture DR, Code Resse (part 1) 12/23	
CNS OTF OF UNO h	Remember ret Instruction	Note:
	ode pointer from stack placed by call instruction ayload	

Notes	
Notes	
Notes	
Notes	
Notes	

CSE Dep. ACS, UPB Lecture 08, Code Reuse (part 1) 14/23

		•
\sim	NI	0
u	IV	\circ

pop-ret

▶	a sequ	uence:	data +	code	pointer	on	the	stack	is	used	by	а
	:gog	ret s	eauence									

- e.g.: pop eax; ret: place data in eax and pop instruction pointer from stack
- \blacktriangleright you may use pop2-ret or pop3-ret etc.

CSE Dep. ACS, UI

Lecture 08, Code Reuse (part 1)

15/23

CNS

Freeing the Stack

- ▶ chain together multiple functions
- ► after calling a function do a pop-ret or popX-ret to free function arguments

CSE Dep. ACS, UPE

Lecture 08, Code Reuse (part 1)

16/23

CNS

ROP Payloads

- offset*"A" + p32(puts) + p32(pop_ret_gadget) +
 p32(puts_string_address)
- offset*"A" + p32(write) + p32(pop3_ret_gadget) +
 p32(1) + p32(buf) + p32(buf_len)

CSE Dep, ACS, UPB

Lecture 08, Code Reuse (part 1)

17/23

CNS

Return Oriented Programming

- $\,\blacktriangleright\,$ using function calls + ret-based calls to chain together code reuse chunks
- ▶ makes use of *ROP gadgets*
- ▶ is a Turing-complete *language*

Notes

Notes

Notes

Notes

ecture 08, Code Reuse (part 1)

18/23

CNS ROP Gadget	Notes
▶ small sequences ending in ret	
use ROPgadget tool (comes with pwntools)use ropgadget or ropsearch or asmsearch in PEDA	
CSE Dep. ACS, UPB Lecture 68, Code Reser (part 1) 19/23	
CNS ROP Chain	
OTF crunch	Notes
 chain together function calls + ROP gadgets do information leak, rewrites, open sockets, run shells 	
CSE Dep. ACS, UPB Lecture 68, Code Resse (part 1) 29/23	
CNS Use Cases	Notes
leak information: variables, addressesopen shell	
► call mprotect() to disable DEP and then inject shellcode	
CSE Dup, ACS, UPB Lecture 66, Code Rosse (part 1) 21/23	
Section (App., Price, of the Section (App., Control	
CNS COUNTY Keywords	Notes
► DEP	
► ROP chain ► return-to-libc ► ROP chain	
▶ ROP	

CSE Dep. ACS, UPB Lecture 08, Code Reuse (part 1) 22/23



References

Notes

- ▶ https://www Shacham/BH Programming
- ▶ https://tra practical-
- ▶ http://code introduction html
- https://git

w.blackhat.com/presentations/bh-usa-08/	
_US_08_Shacham_Return_Oriented_	
g.pdf	
ailofbits.files.wordpress.com/2010/04/	
rop.pdf	
earcana.com/posts/2013/05/28/	
on-to-return-oriented-programming-rop.	
thub.com/JonathanSalwan/ROPgadget	
Lecture 08, Code Resse (part 1) 23/23	
	Notes
	Notes
	Notes