

Introduction to Reversing and Pwning

David Weinman

HERE ALL AND A

Wind Harrison - 2. 200100 - ELL 16"

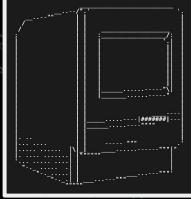
...á…-E -E JM≎œe. á∙⊥tait:

whoami - @h3ll_d0g

- security research engineer at synack, TESC alum
- member of the opentoall ctf team
- hippie millennial loves the pacific northwest
 - snowboarder, skateboarder, metal head





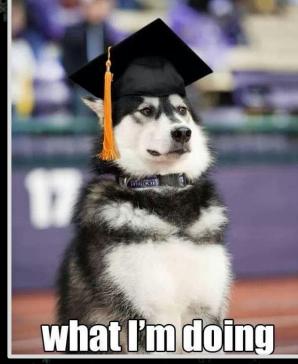




Synack.



i'm still learning I still have no idea





why pwn or reverse?

 amount? not all bugs are equal, not all bounties focus on the same kinds of targets

 synack has a mobile app bounty - can use related skills
 android bugs <= \$4k (google)
 in 2015, average android bug payout was \$2.2k (google)

 fun! pwn challenges can be super rewarding to solve

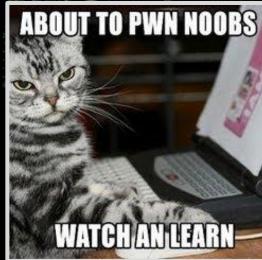


• detailed intro to x86 assembly detailed look at runtime memory layouts overview of common bug classes tooling discussion • glance at mitigations demos along the way

agenda

takeaway

how a program is compiled and run
 common bug classes/exploit mitigations
 ability to decompile c programs
 techniques for binary analysis
 exposure to a pwn challenge







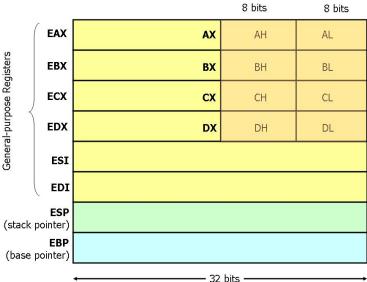




16 bits

X86 / peek inside a CPU x86 arch register: 32 bit memory chunk can refer to 16 bit/8 bit subsets







X86 / assembly language

architecture specific - converted to machine code by assembler
there are two syntax flavors of x86 assembly, <u>intel</u> and at&t
<u>intel syntax</u>: operand destination, source

mov eax, 5 squar	re(int):	
int coupro(int num) {	push mov	ebp ebp, esp
<pre>int square(int num) { return num * num; }</pre>	mov imul	<pre>eax, DWORD PTR [ebp+8] eax, DWORD PTR [ebp+8]</pre>
	pop ret	ebp

Synack.	
	THE NUMBER OF THE PARTY OF THE
D. L. L. K. C.	
alita u culo Mana u culo Mana u culo Mana u culo Angeneration	pôgup or at "; vuo matema uta autora de la composición de la composicinte de la composición de la composición de la comp
} ₩₽₽ ₽₩	au îŭ âu ga== acta da mar 1930ekiijBakoŭ ŭ-acta,ete,ete
	Bight mkt sore to sea
n	generatives telekoeben
hii 64©mà−á C invpinxc í† Binao <i>†</i>)Éz Cme>y*DÜB1	H-SAMSA-44 ;'AçătS…þicen;"iujxà£âl@wîlènewo …?.;; (":Dim\$g\$ZZHNNENEC'a∩ta ZHAN ^{CIII} II -Co gaeriewy £zZaNBIUVd_gMa°*x Tam!OJZ#'`iAlikuç_a"şk/C/ BYG£ ,ILY‡QX™
initi, [°] qëspusiti ∷telëohy ! NBY∢™,	
c::::::::::::::::::::::::::::::::::::	
ang Nataragan La 1251 - La 20 La 1261 - La 12 La 1261 - La	「「「「「「「」」」」」。「「」」」」、「「」」」」、「」」」」、「」」」、「
	上)a#930-× 陳子曼伽甸の設施定冊のX、otà0; t <u>能士[] 坤の</u> 所mLB%- Sr



runtime / heap vs stack

heap is for allocated data usually of variable size, accessible to threads and shared libraries (malloc/free) stack is for local variables/arguments, environment variables and function call metadata

	Low addresses	.text (code)
		.rodata
lf memory layout >	Ling Ling Ling Ling Ling Ling Ling Ling	.data
	YeoBull	heap
	igh addresses	stack



runtime / stack frame layout

caller pushes args and eip onto the stackcallee pushes ebp and local vars onto the stack

low addrs	var2
oid function(int arg1, int arg2) {	var1
<pre>int var1; int var2;</pre>	saved ebp
<pre>void main() {</pre>	saved ret addr
<pre>function(1, 2);</pre>	arg1
high addrs	arg2



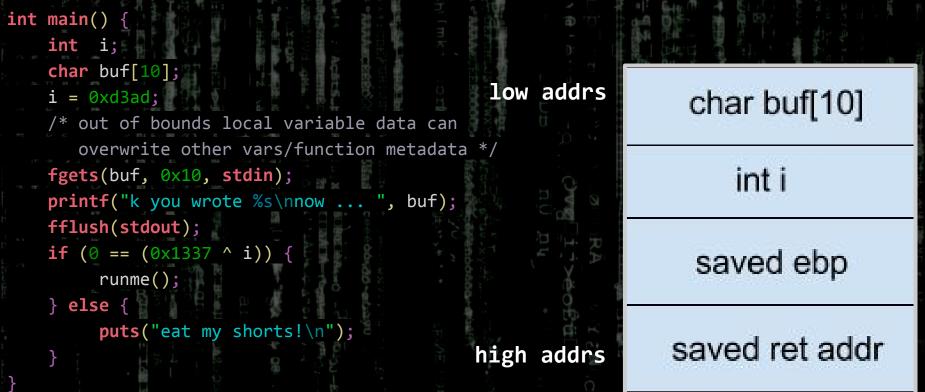
Crackme challenge

shall we play a game?

K	第二十四日の「「「「「「「「「「「」」」」」」「「「「「」」」」」」」」」」」」」」」
ac	
Sy∩	 Martin Martin Marti Martin Martin Martin Martin Martin Martin Martin Martin Mart
	⁴¹ A + シナゆ、身晶ビ「N明 Ammまん回動加船3~」でWebMEFます。 を発
Dere D	
	Care 12 20 10 10 10 10 10 10 10 10 10 10 10 10 10
1 The state	DOGLD O OVELISA AT SUCY
NEDH:	stin of ±=υθ δύ μμ μι out o X
HALLON 13 1. 13 1.	
	B*Ghilmkt iorMac /U 「別たいうを・・・ 「FIME OF
n	je in distrikt friktistur je invationEFGHIUSTURWXYZodetani, oconwxyci, och jgr‴e i jg- <mark>i AAAAwçisEBO</mark> OOO O×XIVE∃BAAz eeeeoon <i>SO×atuby</i> u j
	-SAWSA-MAçistS.,picref"::::Xdificier
i Aqopat: Terdado , terd Acado	BYGE , IUY+QX" ubuquy [®] N-Eline Alizuoy, 20-72012, 21 (Planet [®] by), And OBI 1857 I.COBO MUHARMG [*] Z (AHG [°] EO*SI-
	≈ccsZtati titeteliati letalogianeate statestati dan salendari 3 P
1 - 1 - BH	MART SOLFADE FILLEFILE AND AND ALL POID SELA
. "ಆಗಿಕೊಡಿಯ - ಭಟ್ಟ ಹಾಗಿದ್	(1996年年日本) · · · · · · · · · · · · · · · · · · ·
	LYG#930~× 株 ,他は何の設施工用O_7/**ta0;f%tillimill 把h*ta5。 Sr



bug classes / stack buffer overflow





bug classes / format string

char buf[64]; unsigned int iptr = (unsigned int) &i; disable_buffering(stdout); /* printf takes a variable number of arguments how is the variable known at runtime? */ printf("format string playground, enter buf: "); fgets(buf, 64, stdin); printf("check out the buf you entered: "); printf(buf); if (i) {

printf("congratz u win!\n");
system("/bin/bash");

void main() {

int i = 0;

)./formatstring

format string playground, enter buf: %08x.%08x.%08x check out the buf you entered: 00000040.f774f5a0.080482ba



bug classes / format string

)./formatstring

format string playground, enter buf: %08x.%08x.%08x
check out the buf you entered: 00000040.f774f5a0.080482ba

format string playground, enter buf: %3\$08x check out the buf you entered: 080482ba



bug classes / wild copy

the size of a copy is under limited attacker control
the copy is large enough to cause a fault prior to completing

struct message {
 char buf[24];
 int len;
 void (* result)();

void lose() {
 puts("loser\n");

void win() {
 system("/bin/bash");

void main() {

struct message *msg; disable buffering(stdout); msg = malloc(sizeof(struct message)); $msg \rightarrow len = 24;$ msg->result = lose; while (msg->len >= 24) { printf("What is your length? "); scanf("%d", &msg->len); getc(stdin); // eat up newline printf("OK, what is your buf? "); read(0, msg->buf, msg->len); msg->result(); free(msg);



bug classes / use after free

#define SZ1 20 #define SZ2 15

unsigned int *ptr;

printf("mallocing a chunk\n");

void main() {

malloc chunk is freed but pointer to the chunk is reused
if sizes are similar malloc may reuse chunks after frees

mallocing a chunk
malloced chunk at: 0x09269410
freeing chunk
mallocing chunk again
malloced chunk at: 0x09269410

ptr = malloc(SZ1); printf("malloced chunk at: 0x%08x\n", (unsigned int)ptr); printf("freeing chunk\n"); free(ptr); printf("mallocing chunk again\n"); ptr = malloc(SZ2); printf("malloced chunk at: 0x%08x\n", (unsigned int)ptr); free(ptr);





pwn tools / static & dynamic

ida

static analysis - looking at the binary, its associated shared libraries, anything to do with reversing the binary on disk
dynamic analysis - debugging the process, memory dumping, anything to do with reversing the binary at runtime

	1			[breakp	ints]	[regs:general]
			#1	0x000000100000CF0 h:1 inferior`m	in	[odItsZaPc]
loc_804946A mov eax mov eax test eax	, [ebp+argv]	-97 i e				[RIP: 000000100000CF4 RAX: 000000100000CF6 RBX: 00000000000000000 RBP: 00007FF55FBF986 RDI: 000007FF55FBF986 RDI: 000007FF55FBF986 RDX: 00007FF55FBF986 RDX: 00007FF55FBF986
				hread #1: tid = 0x50c261, 0x000000010		R8 : 0000000000000000
jmp short loc_804949F	■dE loc_8049440: mov eax, [ebp+ mov eax, [eax] mov [esp], eax	a D:	*	frame #0: 0x000000100000cf4 inferio frame #1: 0x00007fff932705c9 libdyld		R9: 00007FF55BFEA48 R10: 0000000000000246 R12: 000000000000000 R13: 00000000000000 R14: 0000000000000 R15: 00000000000000 CS: 002B DS: n/a
	gdb-voltron 🗦	5 80	[ba	cktrace]		ES: n/a FS: 0000 GS: 0000 SS: n/a



pwn tools / binary hacking apis

pwntools (github.com/Gallopsled/pwntools)

[+] Starting local process '/tmp/pwn-a >>> io.sendline("id") >>> io.sendline("exit") >>> print io.recvall() [X] Recieving all data [X] Recieving all data: OB [X] Recieving all data: 60B [+] Recieving all data: Done (60B) [*] Process '/tmp/pwn-asm-LZj2Y0/step3 uid=1000(zerocool) gid=1000(zerocool)



pwn tools / checksec.sh

reveals whether mitigations are on/off

> checksec ./	<u>pwnme</u>
[*] '/home/vo	agrant/talk-stuff/pwnme'
Arch:	i386-32-little
RELRO:	Partial RELRO
Stack:	No canary found
NX:	NX enabled
PIE:	No PIE (0x8048000)

Synck.	
JE MERNINE (IX Z) N.	
WEREPHaim of the ALLENNEY WEREPHaim of the ALLENNEY WEREPHaim of the ALLENNEY	
····································	
hindroma-sahraa.sir ; Aqsts.picry" []]Xaffallelekeno "? 20010-E[] Targanizc'":Dimageszereintesc'ar(z Zharilleatik-rogeneitery";); Brado ; Ez.: Zasriuviges."ax "am! DJZx''iAlk-q-a"ge'C/ De>y*DOBYCE , LLYTQX"	
zinski sistinč (Gh Friktfit distriget og dat distrikter de 	
льный анальный на парада на парада по така. Парада и пара 1997 година – Соландания и парада и пара 1997 година – Соландания и парада и пар	12
F AMABD -1 Ad#930-x # .@dqQBAC#0 .7.•°tà0;# <u>%10.a</u> mabus - y	



wat do / shall we play one last game?

- # prereqs virtualbox & vagrant
- git clone https://github.com/clampz/pwnvm
- vagrant up --provider=virtualbox
- vagrant ssh
- cd challs/intro-examples
- # pwn bins ~/challs/



wat do / bypassing mitigations

I want to go to there.



Pwn challenge





wat do / resources to learn more

- youtube.com/user/GynvaelEN/videos
 liveoverflow.com
- github.com/RPISEC/MBEchallenges.re
- pwnable.tw
 a bug hunter's diary



gr33tz

- synack for sponsoring this talk
- all students involved with RPISEC MBE, especially d00m
- OpenToAll grazfather & uafio for feedback and helping me
- members of cRUcible for inspiration, support, CTFs
- mike_pizza & rweiss for sparking my interest in pwning
- Miguel Gordo Garcia, Robert Musser & inkrypto for testing challs and slides
- Richo Butts for listening to my talk so much i lost count
- BSides crew for arranging everything

