Source Code

int main() {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0	8d 4c 24 04	lea 0x4(%esp),%ecx	0	0	0
4	83 e4 f0	and $0xfffffff0,%esp	0	0	0
7	ff 71 fc	push -0x4(%ecx)	0	0	0
10	55	push %ebp	0	0	0
11	89 e5	mov %esp,%ebp	0	0	0
13	53	push %ebx	0	0	0
14	51	push %ecx	0	0	0
15	83 ec 30	sub $0x30,%esp	0	0	0
18	e8 fc ff ff ff	call 13 <main+0x13>	0	0	0
23	81 c3 02 00 00 00	add $0x2,%ebx	0	0	0
29	65 a1 14 00 00 00	mov %gs: 0x14,%eax	0	0	0
35	89 45 f4	mov %eax,-0xc(%ebp)	0	0	0
38	31 c0	xor %eax,%eax	0	0	0

Source Code

printf("Motivating Example\n");

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
40	83 ec 0c	sub $0xc,%esp	0	0	0
43	8d 83 00 00 00 00	lea 0x0(%ebx), %eax	0	0	0
49	50	push %eax	0	0	0
50	e8 fc ff ff ff	call 33 <main+0x33>	0	0	0

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
55	83 c4 10	add $0x10,%esp	0	0	0

Source Code

int PINSize = 4;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
58	c7 45 d0 04 00 00 00	movl $0x4,-0x30(%ebp)	18.37%	16.33%	20.41%

Source Code

int PINCandidate[] = {1,2,3,4};

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
65	c7 45 d4 01 00 00 00	movl $0x1,-0x2c(%ebp)	0	0	0
72	c7 45 d8 02 00 00 00	movl $0x2,-0x28(%ebp)	0	0	0
79	c7 45 dc 03 00 00 00	movl $0x3,-0x24(%ebp)	0	0	0
86	c7 45 e0 04 00 00 00	movl $0x4,-0x20(%ebp)	0	0	0

Source Code

int PINTrue[] = {4,3,2,1};

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
93	c7 45 e4 04 00 00 00	movl $0x4,-0x1c(%ebp)	0	0	0
100	c7 45 e8 03 00 00 00	movl $0x3,-0x18(%ebp)	0	0	0
107	c7 45 ec 02 00 00 00	movl $0x2,-0x14(%ebp)	0	0	0
114	c7 45 f0 01 00 00 00	movl $0x1,-0x10(%ebp)	0	0	0

Source Code

bool grantAccess = false;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
121	c6 45 ca 00	movb $0x0,-0x36(%ebp)	0	0	0

Source Code

bool badValue = false;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
125	c6 45 cb 00	movb $0x0,-0x35(%ebp)	0	0	0

Source Code

int i = 0;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
129	c7 45 cc 00 00 00 00	movl $0x0,-0x34(%ebp)	46.94%	12.24%	51.02%

Source Code

while (i < PINSize) {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
136	eb 1a	jmp a4 <main+0xa4>	0	0	0

Source Code

if (PINCandidate[i] != PINTrue[i]) {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
138	8b 45 cc	mov -0x34(%ebp), %eax	0	0	0
141	8b 54 85 d4	mov -0x2c(%ebp, %eax,4),%edx	0	0	0
145	8b 45 cc		0	0	0

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
		mov -0x34(%ebp), %eax			
148	8b 44 85 e4	mov -0x1c(%ebp, %eax,4),%eax	0	0	0
152	39 c2	cmp %eax,%edx	11.11%	0	0
154	74 04	je a0 <main+0xa0>	0	0	0

Source Code

badValue = true;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
156	c6 45 cb 01	movb $0x1,-0x35(%ebp)	32.0%	40.0%	40.0%

Source Code

}

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault

Source Code

i++;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
160	83 45 cc 01	addl $0x1,-0x34(%ebp)	0	0	0

Source Code

while (i < PINSize) {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
164	8b 45 cc	mov -0x34(%ebp), %eax	17.65%	5.88%	0
167	3b 45 d0	cmp -0x30(%ebp), %eax	5.88%	11.76%	5.88%
170	7c de	jl 8a <main+0x8a>	0	66.67%	0

Source Code

}

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
Source Code					

if (badValue == false) {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
172	0f b6 45 cb	movzbl -0x35(%ebp),%eax	24.0%	12.0%	8.0%
176	83 f0 01	xor $0x1,%eax	58.82%	17.65%	47.06%
179	84 c0	test %al,%al	33.33%	0	0
181	74 04	je bb <main+0xbb>	0	55.56%	0
Source Code					

grantAccess = true;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
183	c6 45 ca 01	movb $0x1,-0x36(%ebp)	0	0	0
Source Code					

}

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
Source Code					

printf("Grant Access?: %s\n", grantAccess ? "true" : "false");

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
187	80 7d ca 00	cmpb $0x0,-0x36(%ebp)	0	0	0
191	74 08	je c9 <main+0xc9>	0	0	0
193	8d 83 13 00 00 00	lea 0x13(%ebx), %eax	0	0	0

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
199	eb 06	jmp cf <main+0xcf>	0	0	0
201	8d 83 18 00 00 00	lea 0x18(%ebx), %eax	0	0	0
207	83 ec 08	sub $0x8,%esp	0	0	0
210	50	push %eax	0	0	0
211	8d 83 1e 00 00 00	lea 0x1e(%ebx), %eax	0	0	0
217	50	push %eax	0	0	0
218	e8 fc ff ff ff	call db <main+0xdb>	0	0	0
223	83 c4 10	add $0x10,%esp	0	0	0

Source Code

printf("Bad Value?: %s\n", badValue ? "true" : "false");

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
226	80 7d cb 00	cmpb $0x0,-0x35(%ebp)	0	0	0
230	74 08	je f0 <main+0xf0>	0	0	0
232	8d 83 13 00 00 00	lea 0x13(%ebx), %eax	0	0	0
238	eb 06	jmp f6 <main+0xf6>	0	0	0
240	8d 83 18 00 00 00	lea 0x18(%ebx), %eax	0	0	0
246	83 ec 08	sub $0x8,%esp	0	0	0
249	50	push %eax	0	0	0
250	8d 83 31 00 00 00	lea 0x31(%ebx), %eax	0	0	0
256	50	push %eax	0	0	0
257	e8 fc ff ff ff		0	0	0

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
		call 102 <main+0x102>			
262	83 c4 10	add $0x10,%esp	0	0	0

Source Code

assert(!(grantAccess == true && PINCandidate != PINTrue));

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
265	0f b6 45 ca	movzbl -0x36(%ebp),%eax	0	0	0
269	83 f0 01	xor $0x1,%eax	0	0	0
272	84 c0	test %al,%al	0	0	0
274	75 1c	jne 130 <main+0x130>	0	0	0
276	8d 83 84 00 00 00	lea 0x84(%ebx), %eax	0	0	0
282	50	push %eax	0	0	0
283	6a 1c	push $0x1c	0	0	0
285	8d 83 41 00 00 00	lea 0x41(%ebx), %eax	0	0	0
291	50	push %eax	0	0	0
292	8d 83 50 00 00 00	lea 0x50(%ebx), %eax	0	0	0
298	50	push %eax	0	0	0
299	e8 fc ff ff ff	call 12c <main+0x12c>	0	0	0

Source Code

// assert that logic not broken, err if broken

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault

Source Code

return 0;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
304	b8 00 00 00 00	mov $0x0,%eax	0	0	0

Source Code

}

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
309	8b 55 f4	mov -0xc(%ebp), %edx	0	0	0
312	65 2b 15 14 00 00 00	sub %gs: 0x14,%edx	0	0	0
319	74 05	je 146 <main+0x146>	0	0	0
321	e8 fc ff ff ff	call 142 <main+0x142>	0	0	0
326	8d 65 f8	lea -0x8(%ebp), %esp	0	0	0
329	59	pop %ecx	0	0	0
330	5b	pop %ebx	0	0	0
331	5d	pop %ebp	0	0	0
332	8d 61 fc	lea -0x4(%ecx), %esp	0	0	0
335	c3	ret	0	0	0
0	8b 1c 24	mov (%esp),%ebx	0	0	0
3	c3	ret	0	0	0