Source Code					
int main() {					

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0x3c	8d 4c 24 04	lea 0x4(%esp),%ecx	0	0	0
0x40	83 e4 f0	and $0xfffffff0,%esp	0	0	0
0x43	ff 71 fc	push -0x4(%ecx)	0	0	0
0x46	55	push %ebp	0	0	0
0x47	89 e5	mov %esp,%ebp	0	0	0
0x49	53	push %ebx	0	0	0
0x4a	51	push %ecx	0	0	0
0x4b	83 ec 30	sub $0x30,%esp	0	0	0
0x4e	e8 fc ff ff ff	call 13 <main+0x13>	0	0	0
0x53	81 c3 02 00 00 00	add $0x2,%ebx	0	0	0
0x59	65 a1 14 00 00 00	mov %gs:0x14,%eax	0	0	0
0x5f	89 45 f4	mov %eax,-0xc(%ebp)	0	0	0
0x62	31 c0	xor %eax,%eax	0	0	0

Source Code					
int PINSize = 4;					

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0x64	c7 45 d0 04 00 00 00	movl $0x4,-0x30(%ebp)	18.37	16.33	20.41

Source Code					
int PINCandidate[] = {0,0,0,0};					

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0x6b	c7 45 d4 00 00 00 00	movl $0x0,-0x2c(%ebp)	0	0	0
0x72	c7 45 d8 00 00 00 00	movl $0x0,-0x28(%ebp)	0	0	0
0x79	c7 45 dc 00 00 00 00	movl $0x0,-0x24(%ebp)	0	0	0
0x80	c7 45 e0 00 00 00 00	movl $0x0,-0x20(%ebp)	0	0	0

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

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0x87	c7 45 e4 01 00 00 00	movl $0x1,-0x1c(%ebp)	0	0	0

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0x8e	c7 45 e8 02 00 00 00	movl $0x2,-0x18(%ebp)	0	0	0
0x95	c7 45 ec 03 00 00 00	movl $0x3,-0x14(%ebp)	0	0	0
0x9c	c7 45 f0 04 00 00 00	movl $0x4,-0x10(%ebp)	0	0	0

Source Code

bool grantAccess = false;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xa3	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
0xa7	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
0xab	c7 45 cc 00 00 00 00	movl $0x0,-0x34(%ebp)	46.94	12.24	59.18

Source Code

while (i < PINSize) {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xb2	eb 1a	jmp 92 <main+0x92>	0	0	0

Source Code

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

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xb4	8b 45 cc	mov -0x34(%ebp),%eax	0.00	0	0.0
0xb7	8b 54 85 d4	mov -0x2c(%ebp,%eax,4),%edx	0.00	0	0.0
0xbb	8b 45 cc	mov -0x34(%ebp),%eax	0.00	0	0.0
0xbe	8b 44 85 e4	mov -0x1c(%ebp,%eax,4),%eax	0.00	0	0.0
0xc2	39 c2	cmp %eax,%edx	11.11	0	0.0
0xc4	74 04	je 8e <main+0x8e>	0.00	0	0.0

Source Code

badValue = true;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xc6	c6 45 cb 01	movb $0x1,-0x35(%ebp)	32.0	40.0	40.0

Source Code

}

Source Code

i++;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xca	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
0xce	8b 45 cc	mov -0x34(%ebp),%eax	11.76	0.00	11.76
0xd1	3b 45 d0	cmp -0x30(%ebp),%eax	5.88	11.76	29.41
0xd4	7c de	jl 78 <main+0x78>	0.00	66.67	0.00

Source Code

}

Source Code

if (badValue == false) {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xd6	0f b6 45 cb	movzbl -0x35(%ebp), %eax	24.00	12.00	8.00
0xda	83 f0 01	xor $0x1,%eax	58.82	17.65	52.94
0xdd	84 c0	test %al,%al	33.33	0.00	0.00
0xdf	74 04	je a9 <main+0xa9>	0.00	55.56	0.00

Source Code

grantAccess = true;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xe1	c6 45 ca 01	movb $0x1,-0x36(%ebp)	0	0	0

Source Code

}

Source Code

if (grantAccess) {

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xe5	80 7d ca 00	cmpb $0x0,-0x36(%ebp)	0	0	0
0xe9	74 14	je c3 <main+0xc3>	0	0	0

Source Code
printf("Access Granted");

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xeb	83 ec 0c	sub $0xc,%esp	0	0	0
0xee	8d 83 00 00 00 00	lea 0x0(%ebx),%eax	0	0	0
0xf4	50	push %eax	0	0	0
0xf5	e8 fc ff ff ff	call ba <main+0xba>	0	0	0
0xfa	83 c4 10	add $0x10,%esp	0	0	0
0xfd	eb 12	jmp d5 <main+0xd5>	0	0	0

Source Code
} else {

Source Code
printf("Access Denied");

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0xff	83 ec 0c	sub $0xc,%esp	0	0	0
0x102	8d 83 0f 00 00 00	lea 0xf(%ebx),%eax	0	0	0
0x108	50	push %eax	0	0	0
0x109	e8 fc ff ff ff	call ce <main+0xce>	0	0	0
0x10e	83 c4 10	add $0x10,%esp	0	0	0

Source Code
}

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

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0x111	0f b6 45 ca	movzbl -0x36(%ebp), %eax	0.00	4.00	0.00
0x115	83 f0 01	xor $0x1,%eax	52.94	11.76	47.06
0x118	84 c0	test %al,%al	0.00	0.00	0.00
0x11a	75 1c	jne fc <main+0xfc>	0.00	55.56	0.00
0x11c	8d 83 70 00 00 00	lea 0x70(%ebx),%eax	0.00	0.00	0.00
0x122	50	push %eax	0.00	0.00	0.00
0x123	6a 1c	push $0x1c	0.00	0.00	0.00
0x125	8d 83 1d 00 00 00	lea 0x1d(%ebx),%eax	0.00	0.00	0.00
0x12b	50	push %eax	0.00	0.00	0.00
0x12c		lea 0x3c(%ebx),%eax	0.00	0.00	0.00

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
	8d 83 3c 00 00 00				
0x132	50	push %eax	0.00	0.00	0.00
0x133	e8 fc ff ff ff	call f8 <main+0xf8>	0.00	0.00	0.00

Source Code

return 0;

Offset	Bytecode	Assembly Instruction	Set Fault	Reset Fault	Flip Fault
0x138	b8 00 00 00 00	mov $0x0,%eax	0	0	0
0x13d	8b 55 f4	mov -0xc(%ebp),%edx	0	0	0
0x140	65 2b 15 14 00 00 00	sub %gs:0x14,%edx	0	0	0
0x147	74 05	je 112 <main+0x112>	0	0	0
0x149	e8 fc ff ff ff	call 10e <main+0x10e>	0	0	0
0x14e	8d 65 f8	lea -0x8(%ebp),%esp	0	0	0
0x151	59	pop %ecx	0	0	0
0x152	5b	pop %ebx	0	0	0
0x153	5d	pop %ebp	0	0	0
0x154	8d 61 fc	lea -0x4(%ecx),%esp	0	0	0
0x157	c3	ret	0	0	0
0x3c	8b 1c 24	mov (%esp),%ebx	0	0	0
0x3f	c3	ret	0	0	0