

Software Security, Implementation Flaws, and Memory Safety





#293 HRE-THR 850 1930
ALICE SMITH
COACH

SPECIAL INSTRUX: NONE

000000



#293 HRE-THR 850 1930
ALICE SMITHHHHHHHHHHH
HHACH

SPECIAL INSTRUX: NONE

000000



#293 HRE-THR 850 1930
ALICE SMITH
FIRST

SPECIAL INSTRUX: GIVE
PAX EXTRA CHAMPAGNE.

000000



```
char name[20];  
  
void vulnerable() {  
    gets(name);  
}
```

Note: `gets ( )` reads a line of input from standard input (`stdin`) into a character array. The single input is the name of the character array.

```
char name[20];  
char instrux[80] = "none";  
  
void vulnerable() {  
    gets(name);  
}
```

```
char line[512];  
char command[] = "/usr/bin/finger";  
  
void main() {  
    gets(line);  
    ...  
    execv(command, ...);  
}
```

```
char name[20];  
int  seatinfirstclass = 0;  
  
void vulnerable() {  
    gets(name);  
}
```

```
char name[20];  
int  authenticated = 0;  
  
void vulnerable() {  
    gets(name);  
}
```

```
void vulnerable() {  
    char buf[64];  
    gets(buf);  
    ...  
}
```

```
void still_vulnerable() {  
    char buf = malloc(64);  
    gets(buf);  
    ...  
}
```

Why? How is this different from the previous?

```
void safe() {  
    char buf[64];  
    fgets(buf, sizeof buf, stdin);  
    ...  
}
```

```
void vulnerable() {  
    char buf[64];  
    if (fgets(buf, 64, stdin) == NULL)  
        return;  
    printf(buf);  
}
```

Note: `fgets()` returns null if there is a read error, or if it reaches the end of the file (user types something like control-Z) *before* any characters have been read.

```
void vulnerable(int len, char *data) {  
    char buf[64];  
    if (len > 64)  
        return;  
    memcpy(buf, data, len);  
}
```

Why is this vulnerable? How do we make it safe?

```
void vulnerable(size_t len, char *data) {  
    char *buf = malloc(len+2);  
    memcpy(buf, data, len);  
    buf[len] = '\n';  
    buf[len+1] = '\0';  
}
```

Why is this vulnerable? How do we make it safe?

Table 1: Overall Results

Rank	Flaw	TOTAL	2001	2002	2003	2004	2005	2006
Total		18809	1432	2138	1190	2546	4559	6944
[1]	XSS	13.8%	02.2% (11)	08.7% (2)	07.5% (2)	10.9% (2)	16.0% (1)	18.5% (1)
		2595	31	187	89	278	728	1282
[2]	buf	12.6%	19.5% (1)	20.4% (1)	22.5% (1)	15.4% (1)	09.8% (3)	07.8% (4)
		2361	279	436	268	392	445	541
[3]	sql-inject	09.3%	00.4% (28)	01.8% (12)	03.0% (4)	05.6% (3)	12.9% (2)	13.6% (2)
		1754	6	38	36	142	588	944
[4]	php-include	05.7%	00.1% (31)	00.3% (26)	01.0% (13)	01.4% (10)	02.1% (6)	13.1% (3)
		1065	1	7	12	36	96	913
[5]	dot	04.7%	08.9% (2)	05.1% (4)	02.9% (5)	04.2% (4)	04.3% (4)	04.5% (5)
		888	127	110	34	106	196	315
[6]	infoleak	03.4%	02.6% (9)	04.2% (5)	02.8% (6)	03.8% (5)	03.8% (5)	03.1% (6)
		646	37	89	33	98	175	214
[7]	dos-malform	02.8%	04.8% (3)	05.2% (3)	02.5% (8)	03.4% (6)	01.8% (8)	02.0% (7)
		521	69	111	30	86	83	142
[8]	link	01.8%	04.5% (4)	02.1% (9)	03.5% (3)	02.8% (7)	01.9% (7)	00.4% (16)
		341	64	45	42	72	87	31
[9]	format-string	01.7%	03.2% (7)	01.8% (10)	02.7% (7)	02.4% (8)	01.7% (9)	00.9% (11)
		317	46	39	32	62	76	62
[10]	crypt	01.5%	03.8% (5)	02.7% (6)	01.5% (9)	00.9% (16)	01.5% (10)	00.8% (13)
		278	55	58	18	22	69	56

IE's Role in the Google-China War



By Richard Adhikari
TechNewsWorld
01/15/10 12:25 PM PT

The hack attack on Google that set off the company's ongoing standoff with China appears to have come through a zero-day flaw in Microsoft's Internet Explorer browser. Microsoft has released a security advisory, and researchers are hard at work studying the exploit. The attack appears to consist of several files, each a different piece of malware.

Computer security companies are scurrying to cope with the fallout from the Internet Explorer (IE) flaw that led to cyberattacks on [Google](#) (Nasdaq: GOOG) and its corporate and individual customers.

The zero-day attack that exploited IE is part of a lethal cocktail of malware that is keeping researchers very busy.

"We're discovering things on an up-to-the-minute basis, and we've seen about a dozen files dropped on infected PCs so far," Dmitri Alperovitch, vice president of research at [McAfee](#) Labs, told TechNewsWorld.

The attacks on Google, which appeared to originate in China, have sparked a feud between the Internet giant and the nation's government over censorship, and it could result in Google pulling away from its business dealings in the country.

Pointing to the Flaw

The vulnerability in IE is an invalid pointer reference, [Microsoft](#) (Nasdaq: MSFT) said in [security advisory 979352](#), which it issued on Thursday. Under certain conditions, the invalid pointer can be accessed after an object is deleted, the advisory states. In specially crafted attacks, like the ones launched against Google and its customers, IE can allow remote execution of code when the flaw is exploited.

Broward Vote-Counting Blunder Changes Amendment Result

POSTED: 1:34 pm EST November 4, 2004

BROWARD COUNTY, Fla. -- The Broward County Elections Department has egg on its face today after a computer glitch misreported a key amendment race, according to WPLG-TV in Miami.

Amendment 4, which would allow Miami-Dade and Broward counties to hold a future election to decide if slot machines should be allowed at racetracks, was thought to be tied. But now that a computer glitch for machines counting absentee ballots has been exposed, it turns out the amendment passed.

"The software is not geared to count more than 32,000 votes in a precinct. So what happens when it gets to 32,000 is the software starts counting backward," said Broward County Mayor Ilene Lieberman.

That means that Amendment 4 passed in Broward County by more than 240,000 votes rather than the 166,000-vote margin reported Wednesday night. That increase changes the overall statewide results in what had been a neck-and-neck race, one for which recounts had been going on today. But with news of Broward's error, it's clear amendment 4 passed.



Broward County Mayor Ilene Lieberman says voting counting error is an "embarrassing mistake."