Overflows, Injection, and Memory Safety

CS 161: Computer Security

Prof. David Wagner

January 22, 2016

Announcements

- Midterm 1 moved to Wednesday, Feb 24, 8:00-9:30pm
- Enrollment has been increased (yay!)
- Discussion section sign-ups will be posted this weekend – watch Piazza, and sign up for sections online

Traveler Information

Traveler 1 - Adults (age 18 to 64)

To comply with the TSA Secure Flight program, the traveler information listed here must exactly match the information on the government-issued photo ID that the traveler presents at the airport.

Title (optional): Dr.

First Name: Alice

Middle Name:

Last Name: Smith

Travelers are required to enter a middle name/initial if one is listed on their government-issued photo ID.

Gender: Female

Date of Birth: 01/24/93

Some younger travelers are not required to present an ID when traveling within the U.S. Learn more

+ **Known Traveler Number/Pass ID (optional):**

+ **Redress Number (optional):**

Seat Request:
(•) No Preference ( ) Aisle ( ) Window


#293 HRE-THR 850 1930
ALICE SMITH
COACH
SPECIAL INSTRUX: NONE

Traveler Information

Traveler 1 - Adults (age 18 to 64)

To comply with the TSA Secure Flight program, the traveler information listed here must exactly match the information on the government-issued photo ID that the traveler presents at the airport.

Title (optional): Dr.

First Name: Alice

Middle Name:

Last Name: Smithhhhhhhhhhhhh

Travelers are required to enter a middle name/initial if one is listed on their government-issued photo ID.

Gender: Female

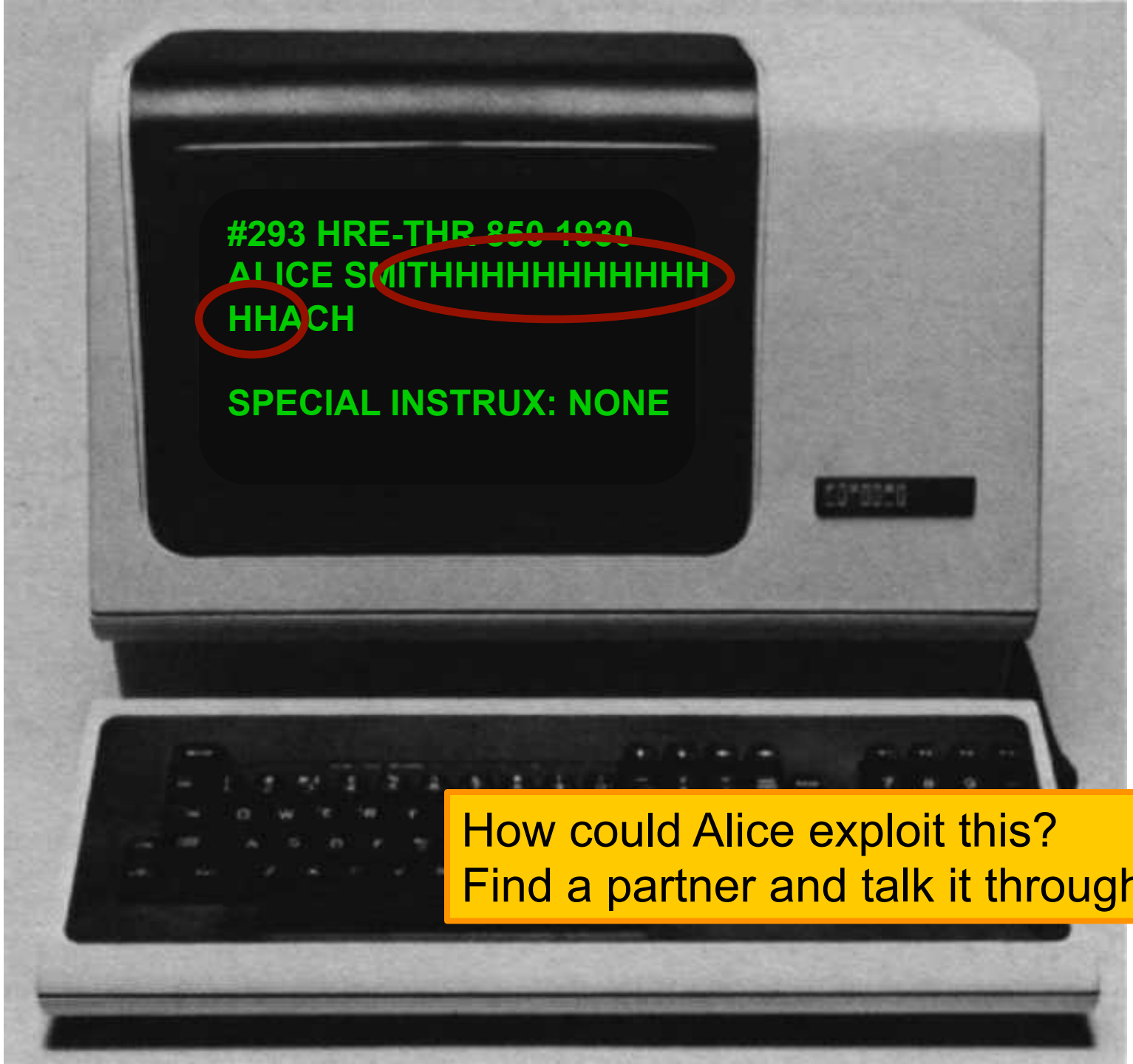
Date of Birth: 01/24/93

Some younger travelers are not required to present an ID when traveling within the U.S. Learn more

+ **Known Traveler Number/Pass ID (optional):**

+ **Redress Number (optional):**

Seat Request:
(•) No Preference ( ) Aisle ( ) Window


#293 HRE-THR 850 1930
ALICE SMITHHHHHHHHHHH
HHACH
SPECIAL INSTRUX: NONE
How could Alice exploit this?
Find a partner and talk it through.

Traveler Information

Traveler 1 - Adults (age 18 to 64)

To comply with the TSA Secure Flight program, the traveler information listed here must exactly match the information on the government-issued photo ID that the traveler presents at the airport.

Title (optional): Dr.

First Name: Alice

Middle Name:

Last Name: Smith First

Travelers are required to enter a middle name/initial if one is listed on their government-issued photo ID.

Gender: Female

Date of Birth: 01/24/93

Some younger travelers are not required to present an ID when traveling within the U.S. Learn more

+ **Known Traveler Number/Pass ID (optional):** ?

+ **Redress Number (optional):** ?

Seat Request:
(•) No Preference ( ) Aisle ( ) Window


#293 HRE-THR 850 1930
ALICE SMITH
FIRST
SPECIAL INSTRUX: NONE


#293 HRE-THR 850 1930
ALICE SMITH
FIRST
SPECIAL INSTRUX: GIVE
PAX EXTRA CHAMPAGNE.

```
char name[20];

void vulnerable() {
   ...
   gets(name);
   ...
}
```

```
char name[20];
char instrux[80] = "none";

void vulnerable() {
   ...
   gets(name);
   ...
}
```

DEMO

```
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 (*fnptr)();

void vulnerable() {
   ...
   gets(name);
   ...
}
```

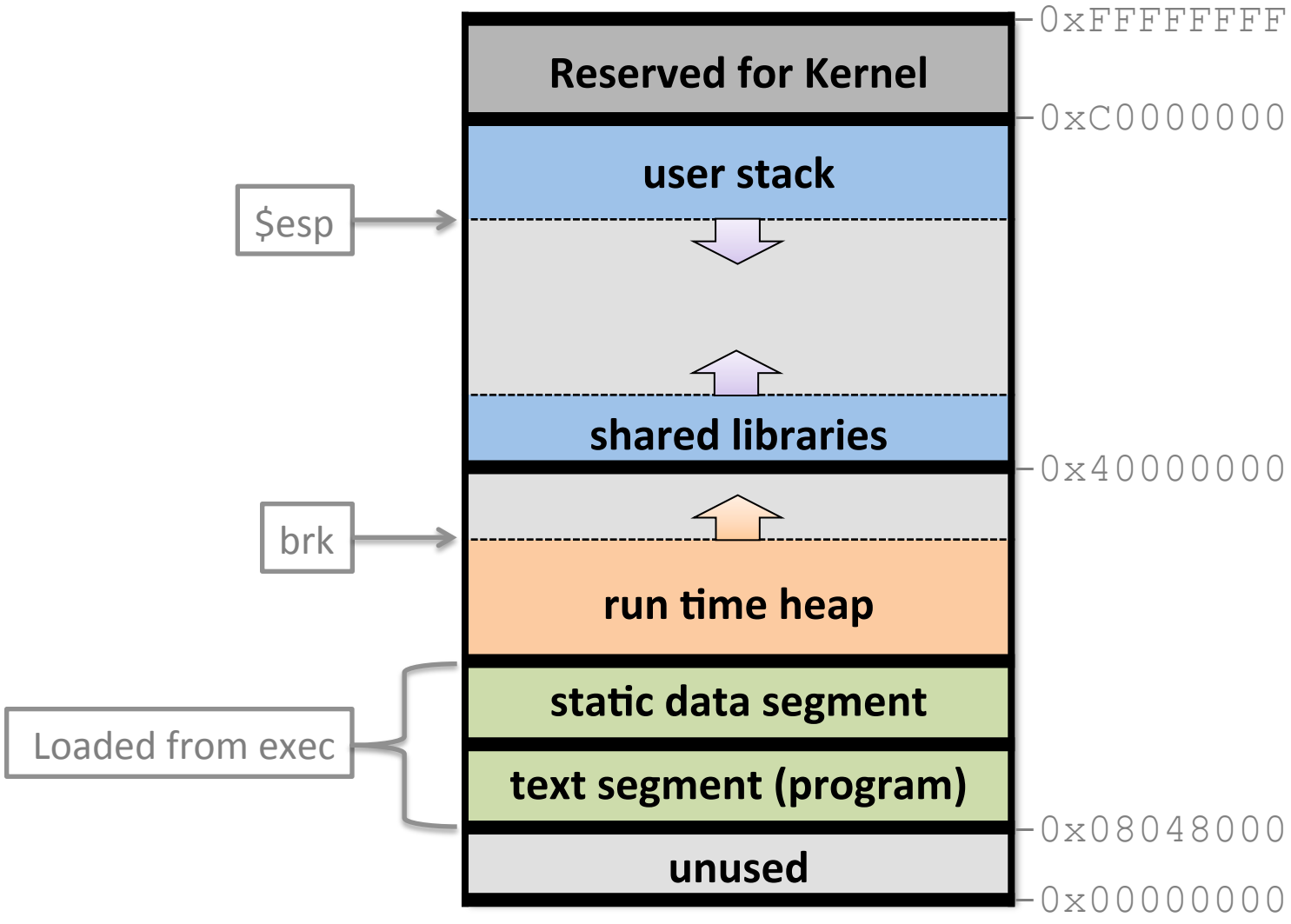
```
char name[20];
int  seatinfirstclass = 0;

void vulnerable() {
   ...
   gets(name);
   ...
}
```

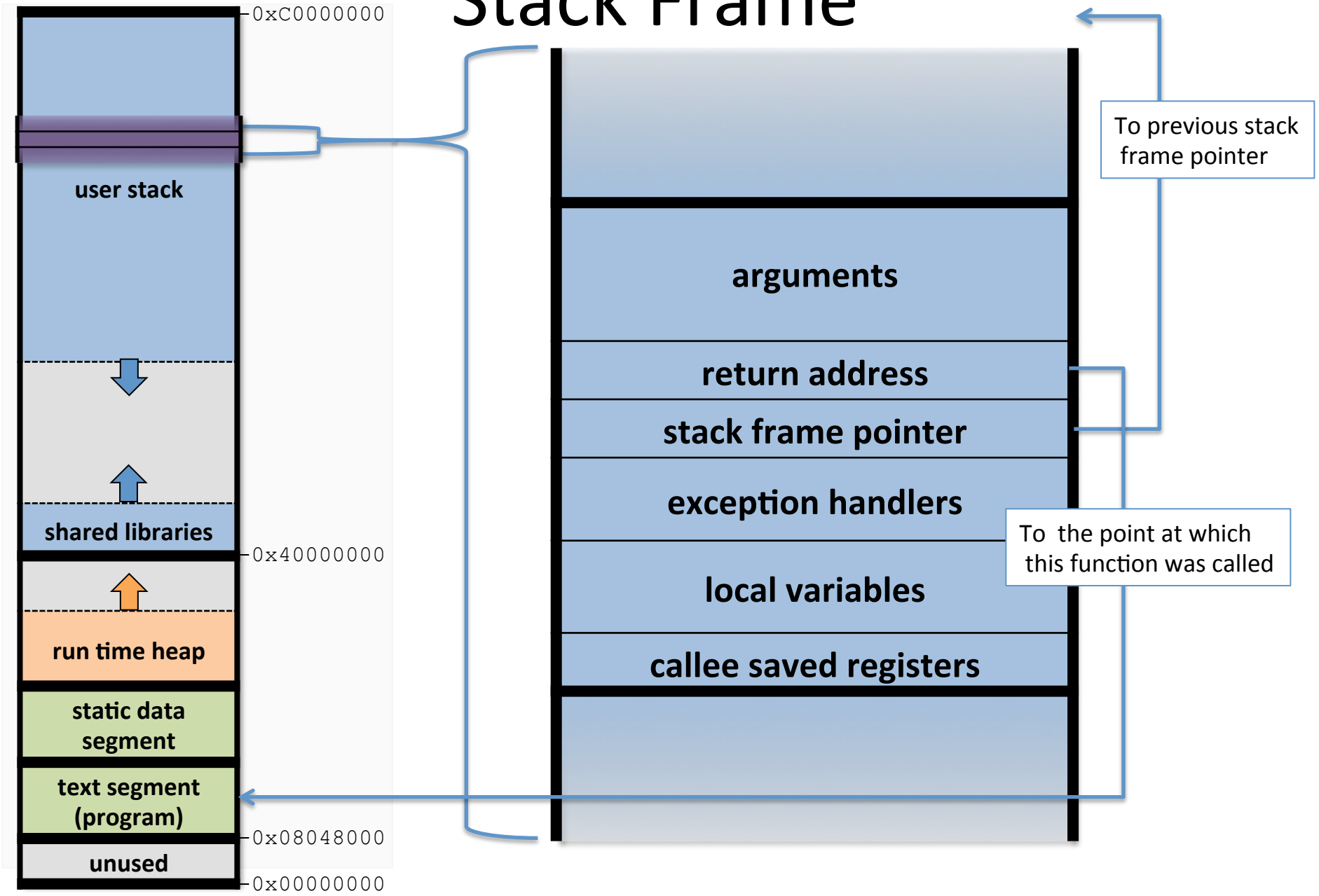
```
char name[20];
int  authenticated = 0;

void vulnerable() {
   ...
   gets(name);
   ...
}
```

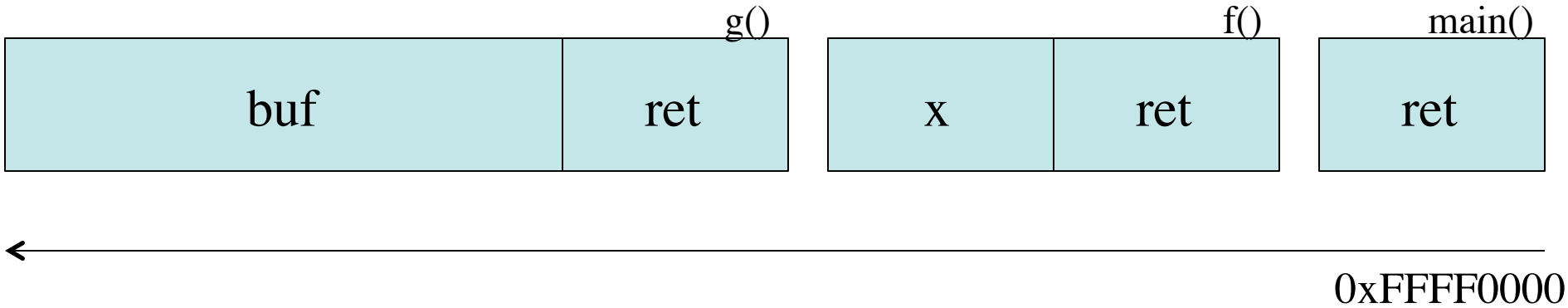
Linux (32-bit) process memory layout



Stack Frame


0xC0000000
user stack
shared libraries
0x40000000
run time heap
static data segment
text segment (program)
0x08048000
unused
0x00000000
arguments
return address
stack frame pointer
exception handlers
local variables
callee saved registers
To previous stack frame pointer
To the point at which this function was called

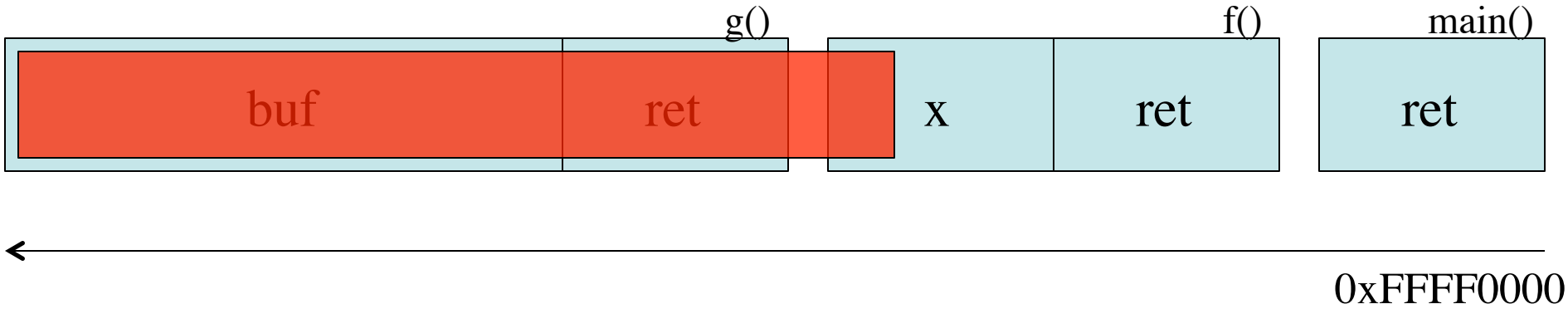
Code Injection




```
main() {
  f();
}
```

```
f() {
  int x;
  g();
}
```

```
g() {
  char buf[80];
  gets(buf);
}
```




```
main() {
  f();
}
```

```
f() {
  int x;
  g();
}
```

```
g() {
  char buf[80];
  gets(buf);
}
```

Basic Stack Exploit

- Overwriting the return address allows an attacker to redirect the flow of program control.
- Instead of crashing, this can allow *arbitrary* code to be executed.
- Example: attacker chooses malicious code he wants executed (“shellcode”), compiles to bytes, includes this in the input to the program so it will get stored in memory somewhere, then overwrites return address to point to it.

Rank	Score	ID	Name
[1]	93.8	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
[2]	83.3	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
[3]	79.0	CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')
[4]	77.7	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[5]	76.9	CWE-306	Missing Authentication for Critical Function
[6]	76.8	CWE-862	Missing Authorization
[7]	75.0	CWE-798	Use of Hard-coded Credentials
[8]	75.0	CWE-311	Missing Encryption of Sensitive Data
[9]	74.0	CWE-434	Unrestricted Upload of File with Dangerous Type
[10]	73.8	CWE-807	Reliance on Untrusted Inputs in a Security Decision
[11]	73.1	CWE-250	Execution with Unnecessary Privileges
[12]	70.1	CWE-352	Cross-Site Request Forgery (CSRF)
[13]	69.3	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
[14]	68.5	CWE-494	Download of Code Without Integrity Check
[15]	67.8	CWE-863	Incorrect Authorization
[16]	66.0	CWE-829	Inclusion of Functionality from Untrusted Control Sphere

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

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

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.

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

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

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

```
memcpy(void *s1, const void *s2, size_t n);
```

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

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

Is it safe? Talk to your partner.

Vulnerable!
If `len = 0xffffffff`, *allocates only 1 byte*

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.

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

"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.

BONUS FOR THE BORED

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

Fun With `printf` Format Strings …

```
printf("100% dude!");
```

⇒ prints value 4 bytes above retaddr as integer

```
printf("100% sir!");
```

⇒ prints bytes pointed to by that stack entry up through first NUL

```
printf("%d %d %d %d ...");
```

⇒ prints series of stack entries as integers

```
printf("%d %s");
```

⇒ prints value 4 bytes above retaddr plus bytes pointed to by preceding stack entry

```
printf("100 % nuke'm!");
```

⇒ **writes** the value 3 to address pointed to by stack entry