Overflows, Injection, and Memory Safety

CS 161: Computer Security
Prof. Vern Paxson

TAs: Jethro Beekman, Mobin Javed, Antonio Lupher, Paul Pearce & Matthias Vallentin

http://inst.eecs.berkeley.edu/~cs161/

January 24, 2013
## Traveler Information

### Traveler 1 - Adults (age 18 to 64)

To comply with the [TSA Secure Flight program](https://www.tsa.gov), the traveler information listed here must exactly match the information on the government-issued photo ID that the traveler presents at the airport.

<table>
<thead>
<tr>
<th>Title (optional):</th>
<th>First Name:</th>
<th>Middle Name:</th>
<th>Last Name:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dr. Alice</td>
<td></td>
<td></td>
<td>Smith</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gender:</th>
<th>Date of Birth:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>01/24/93</td>
</tr>
</tbody>
</table>

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

Some younger travelers are not required to present an ID when traveling within the U.S. [Learn more](https://www.tsa.gov/travel/identity-documentation)

**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](https://www.tsa.gov), the traveler information listed here must exactly match the information on the government-issued photo ID that the traveler presents at the airport.

<table>
<thead>
<tr>
<th>Title (optional):</th>
<th>First Name:</th>
<th>Middle Name:</th>
<th>Last Name:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dr.</td>
<td>Alice</td>
<td></td>
<td>Smithhhhhhhhhhhhh</td>
</tr>
</tbody>
</table>

**Gender:** Female

**Date of Birth:** 01/24/93

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

Some younger travelers are not required to present an ID when traveling within the U.S. [Learn more](https://www.tsa.gov/travel/children-and-young-adults).

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

**Redress Number (optional):**

**Seat Request:**
- [ ] No Preference
- [ ] Aisle
- [ ] Window
#293 HRE-THR 850 1930
ALICE SMITHHHHHHHHHHHHHH
HHACH

SPECIAL INSTRUX: NONE
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

Gender: Female  Date of Birth: 01/24/93

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

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);
    ...
}
char line[512];
char command[] = "/usr/bin/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);
    ...
}


Code Injection

Illustrations courtesy Matthias Vallentin
<table>
<thead>
<tr>
<th>Rank</th>
<th>Score</th>
<th>ID</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>93.8</td>
<td>CWE-89</td>
<td>Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')</td>
</tr>
<tr>
<td>[2]</td>
<td>83.3</td>
<td>CWE-78</td>
<td>Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')</td>
</tr>
<tr>
<td>[3]</td>
<td>79.0</td>
<td>CWE-120</td>
<td>Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')</td>
</tr>
<tr>
<td>[4]</td>
<td>77.7</td>
<td>CWE-79</td>
<td>Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')</td>
</tr>
<tr>
<td>[6]</td>
<td>76.8</td>
<td>CWE-862</td>
<td>Missing Authorization</td>
</tr>
<tr>
<td>[7]</td>
<td>75.0</td>
<td>CWE-798</td>
<td>Use of Hard-coded Credentials</td>
</tr>
<tr>
<td>[8]</td>
<td>75.0</td>
<td>CWE-311</td>
<td>Missing Encryption of Sensitive Data</td>
</tr>
<tr>
<td>[9]</td>
<td>74.0</td>
<td>CWE-434</td>
<td>Unrestricted Upload of File with Dangerous Type</td>
</tr>
<tr>
<td>[10]</td>
<td>73.8</td>
<td>CWE-807</td>
<td>Reliance on Untrusted Inputs in a Security Decision</td>
</tr>
<tr>
<td>[11]</td>
<td>73.1</td>
<td>CWE-250</td>
<td>Execution with Unnecessary Privileges</td>
</tr>
<tr>
<td>[12]</td>
<td>70.1</td>
<td>CWE-352</td>
<td>Cross-Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>[13]</td>
<td>69.3</td>
<td>CWE-22</td>
<td>Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')</td>
</tr>
<tr>
<td>[14]</td>
<td>68.5</td>
<td>CWE-494</td>
<td>Download of Code Without Integrity Check</td>
</tr>
<tr>
<td>[16]</td>
<td>66.0</td>
<td>CWE-829</td>
<td>Inclusion of Functionality from Untrusted Control Sphere</td>
</tr>
</tbody>
</table>
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()
{
  char buf[64];
  if (fgets(buf, 64, stdin) == NULL)
    return;
  printf(buf);
}
Fun With printf Format Strings ...

printf("100% dude!");
  ⇒ prints stack entry 4 bytes above RIP 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 stack entry 4 bytes above RIP plus bytes
      pointed to by preceding stack entry
printf("100% nuke’m!");
  ⇒ writes the value 3 to address pointed
       to by stack entry
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 vulnerable(size_t len, char *data) {
    char *buf = malloc(len+2);
    if (buf == 0) return;
    memcpy(buf, data, len);
    buf[len] = '\n';
    buf[len+1] = '\0';
}

If \textit{len} = \texttt{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.

"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.
Questions?