Secure Programming

An introduction to Splint

Christian D. Jensen René Rydhof Hansen

Informatics and Mathematical Modelling
Technical University of Denmark

E05-02230

The Problem

Program bugs primary attack vector

- The Internet Worm (November 1988)
- ...

Bad programming

- Buffer overflow
- Race conditions (TOCTTOU)
- Dereferencing `null`-pointers
- Use before def.
- ...

Buffer Overflow

What is a buffer overflow?

- Read/write beyond memory allocated to buffer
  - Unchecked user input
  - Unchecked environment variables
  - Filenames assumed to be sane
  - Assuming network packets are well-formed
  - ...
- May overwrite return addresses (e.g., stack overflow)
- May insert jumps to library code (e.g., heap overflow)

Beware

- Account for 50+% of reported vulnerabilities [Larochelle 2001]
- Very hard to find/avoid

Buffer Overflow

Example

```
void  updateEnv(char *str)
{
  char *tmp;

  tmp = getenv("HOME");  /* <-- No length limit */
  if (tmp != NULL)
    strcpy(str,tmp);     /* <-- Use strncpy */

}
```

Buffer Overflow

Example

```
void  updateEnv(char *str)
{
  char *tmp;

  tmp = getenv("HOME");  /* <-- No length limit */
  if (tmp != NULL)
    strcpy(str,tmp);     /* <-- Use strncpy */

}
```

Buffer Overflow

Example

```
void  updateEnv(char *str, size_t size)
{
  char *tmp;

  tmp = getenv("HOME");  /* <-- No length limit */

  if (tmp != NULL)
  {
      strncpy(str,tmp,size-1);
      str[size -1] = '\0';
  }
}
```

Time-Of-Check-To-Time-Of-Use (TOCTTOU) Flaws

How does it work?

- Checking file permissions, do something, open the file and write to it... file may have changed permission in-between

Example

```
                        creat("/tmp/X")
access("/tmp/X")
                        unlink("/tmp/X")
                        symlink("/tmp/X","/etc/passwd")
open("/tmp/X")
```

Time-Of-Check-To-Time-Of-Use (TOCTTOU) Flaws

How does it work?

- Checking file permissions, do something, open the file and write to it... file may have changed permission in-between

Example

```
                              creat("/tmp/X")
access("/tmp/X")
                              unlink("/tmp/X")
                              symlink("/tmp/X","/etc/passwd")
open("/tmp/X")
```

De-referencing `null`-pointers

What is it?

- Following a `null`-pointer
  - Trying to use a "freed" pointer
  - Not checking a freshly `malloc`'ed pointer
  - ...

Example

```
void  foo(void)
{
  char *tmp;

  tmp = (char *) malloc(MAXTMP);
  *tmp = 'X';
}
```

De-referencing null-pointers

What is it?

- Following a null-pointer
  - Trying to use a "freed" pointer
  - Not checking a freshly malloc'ed pointer
  - ...

Example

```
void  foo(void)
{
  char *tmp;

  tmp = (char *) malloc(MAXTMP);
  if(tmp != NULL)
    *tmp = 'X';
}
```

Solving the Problem of Bad Programming

Solutions?

- Teach (force?) programmers to be more careful
- Use safe(r) languages
- More and better testing
- Formal methods
- Language-Based Techniques

Language-Based Technology

Using programming language techniques to verify safety and security of programs

Example

Java and C# (bytecode verification, type systems, sandboxing, ...)

Solving the Problem of Bad Programming

Solutions?

- Teach (force?) programmers to be more careful Long-term!
- Use safe(r) languages Sometimes you need C
- More and better testing Cannot cover full program
- Formal methods Time consuming, expensive
- Language-Based Techniques

Language-Based Technology

Using programming language techniques to verify safety and security of programs

Example

Java and C# (bytecode verification, type systems, sandboxing, ...)

Language-Based Technology

Techniques

- Software Model Checking
- Certifying Compilers
- Proof Carrying Code (PCC)
- Inlined Reference Monitors
- Type-Systems
- Static Analysis

Static Analysis

- Roots: optimising compilers
- Static computation of dynamic behaviour
  - Approximation used to sidestep halting-problem

Splint: Light-Weight Annotation-Based Static Analysis

Secure Programming LINT

- Based on `lint`: well-known program checker
- Let the programmer annotate program
- Check that the program is consistent with annotations
- Can find many common errors

Other Tools

- SLAM (used at Microsoft)
- BLAST
- Bandera
- CQUAL
- MOPS
- ...

Splint: De-referencing null-pointers

Example

```
      void  foo(void)
      {
        char *tmp;

        tmp = (char *) malloc(MAXTMP);
        *tmp = 'X';
        free(tmp);
      }
```

```
 $ splint ex01.c
...
ex01.c:8:4: Dereference of possibly null pointer tmp: *tmp
  A possibly null pointer is dereferenced...
```

Splint: De-referencing null-pointers

Example

```
      void  foo(void)
      {
        char *tmp;

        tmp = (char *) malloc(MAXTMP);
        if(tmp != NULL)
          *tmp = 'X';
        free(tmp);
      }
```

```
 $ splint ex01.c
...
Finished checking --- no warnings
```

Splint: Buffer Overflow

Example

```
void  updateEnv(char *str)

{
  char *tmp;

  tmp = getenv("HOME");
  if (tmp != NULL) {
    strcpy(str,tmp);
}

$ splint +bounds buffer01.c
...
buffer01:9: Possible out-of-bounds store: strcpy(str,tmp)...
```

Splint: Buffer Overflow

Example

```
void  updateEnv(char *str, size_t size)
                /*@requires maxSet(str) >= size -1@*/
{
  char *tmp;

  tmp = getenv("HOME");
  if (tmp != NULL) {
    strncpy(str,tmp,size-1);
    str[size -1] = '\0';
  }
}

$ splint +bounds buffer01.c
...
Finished checking --- no warnings
```

Splint: Buffer Overflow

Annotations

- `maxSet(b)`: max. index of `b` that is assigned
- `maxRead(b)`: max. index of `b` that is read

Example

```
void  updateEnv(char *str, size_t size)
     /*@requires maxSet(str) >= size -1@*/
```

- In order for `updateEnv` to "work":
  - Parameter `str` must be "settable" upto (and including) position `size - 1`
  - Note: `str[size -1] = '\0'`

Splint: Summary

Only scratched the surface!

- Cant catch many common programming errors
  - Memory modelling
  - Sharing
  - Control Flow
  - *User defined*

The Downside?

- May need a lot of annotation
- Not complete
- Not sound
- ...