Introduction to Software Security

Software Vulnerabilities (1)

Seong-je Cho

Spring 2020

Computer Security & Operating Systems Lab,

Dept. Software Science, DKU

Sources / References

- *Information Security: Principles and Practice*, 2nd edition by Mark Stamp, Wiley, 2011
- Software Security: Principles, Policies, and Protection, Matias Payer, Apr. 2019.
- N. Vlajic, CSE 3482: Introduction to Computer Security, Yorku
- Nicholas Weaver, Computer Science 161: Computer Security, Berkeley
- Myrto Arapinis, Computer Security: INFRA10067, University of Edinburgh
- Lecture 12 Program Security, CS 450/650 Lecture

Contents

- Why Software Security?
- Examples Software Bugs/Flaws
  - Improper initialization
  - Side effects
  - Scoping
  - Control flows
  - Integer security
  - Null pointer dereference
  - Operator precedence logic error
- Bad software is everywhere!

Software is ubiquitous

- **Systems software :** OS, compiler, loader
- **Business software :** Payroll, accounting
- **Scientific and engineering software**
  - **Computer-aided design, simulation, weather prediction, …**
- **Internet software:**
  - **B2C: business-to-customer (e.g., amazon.com)**
  - **Facebook, Google Chrome, …**
- **PC software :** Spreadsheets, word processing, games, …
- **Embedded software**
  - **Cars, microwave ovens, cable boxes, light switches, “smart dust”, …**
- **Mobile applications**

Why Software Security?

- Why is software as important to security as crypto, access control and protocols?
  - Virtually all of computer security is implemented in software
- If your software is subject to attack, your security is broken
  - Regardless of strength of crypto, access control or protocols
- Software can be a poor foundation for security
  - Software Vulnerability
  - Bug, Flaw, Defect, Weakness, …

Software Crisis?

- People that design and build software have to deal with many problems
- Software crisis for the last 30 years?
  - In reality, things are not that bad
    - Many more successes than failures
    - But problems are persistent
- The SE field is still immature
  - e.g., compare with civil engineering, etc.

☞ Source: (from an old material)
- 198:431 Software Engineering
- Fall 2006, Prof Barbara G. Ryder

Software

Software is:

- Executable programs
- Data associated with these programs
- Documents: user requirements, design documents, user/programmer guides, etc.

Software plays a key role

- Process, deliver, and save data
- Produces, manages, and presents information
- Information society
  - Next step after industrial society

Bugs, Defects, Weaknesses, and Vulnerabilities

- Improper initialization
- Side effects
- Scoping
- Operator precedence
- Divide-by Zero
- Infinite loop
- Type confusion (illegal downcasts)
- Deadlock
- Integer Overflow / Underflow
- Memory leak
  - Use-after-free
- Buffer overflow = Buffer overrun
- Time-of-check-to-time-of-use flaw
- Format string bug

Examples of SW Bugs

```
typedef unsigned int uint;
int getmin(int *arr, uint len) {
  int min;
  for (int i=0; i<len; i++)
    min = (min < arr[i]) ? min : arr[i];
  return min;
}
```

Improper Initialization

Side Effects

bar = ?
baz = ?

```
if (foo == 12 || (bar = 13))
  baz == 12;
```

Examples of SW Bugs

Scoping

a = ?

```
int a;
void calc(int b) {
  int a = b*12;
  if (b + 24 == 96)
    a = b;
}
```

Control-flow (1)

```
int x,y;
for (x=0; x<xlen; x++)
  for (y=0; y<ylen; y++);
    pix[y*xlen + x] = x*y;
```

If xlen=10, ylen=5,
pix[0] = ? pix[1] = ? pix[2] = ?
pix[3] = ?
After the 2nd for statement, x = ?, y = ?

Examples of SW Bugs

Control flow (2)

```
if (isbad(cert))
  goto fail;
if (invalid(cert))
  goto fail;
  goto fail;
```

```
    …
L10: printf("Hello, world\n");
    goto L10;
```

The Apple goto fail vulnerability

```
static OSStatus
SSLVerifySignedServerKeyExchange(SSLContext *ctx, bool isRsa, SSLBuffer
signedParams,
                                 uint8_t *signature, UInt16 signatureLen)
{
    OSStatus        err;
    ...

    if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
        goto fail;
    if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
        goto fail;
        goto fail;
    if ((err = SSLHashSHA1.final(&hashCtx, &hashOut)) != 0)
        goto fail;
    ...

fail:
    SSLFreeBuffer(&signedHashes);
    SSLFreeBuffer(&hashCtx);
    return err;
}
```

Examples of SW Bugs

```
float x = 0.1;
while (x != 1.1) {
   x = x + 0.1;
   printf("x = %f\n", x);
}
```

```
int k = 1
int val = 0;

while (k = 10) {
            val++;
            k++;
}
printf (“k = %d, val = %d \n”, k, val);
```

Control-flow (2) -- Loop

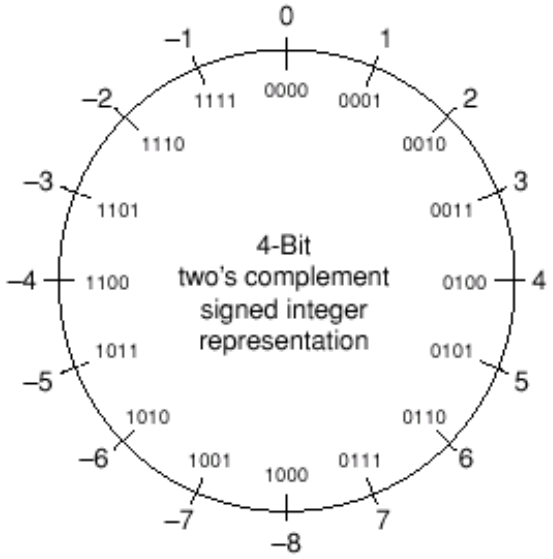
How many times will this loop run?

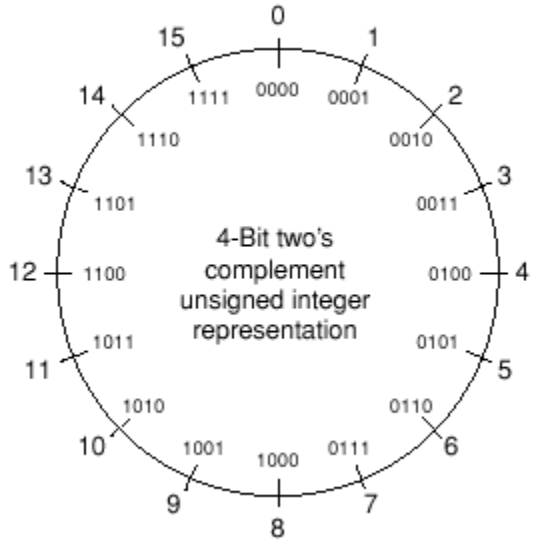
Examples of SW Bugs

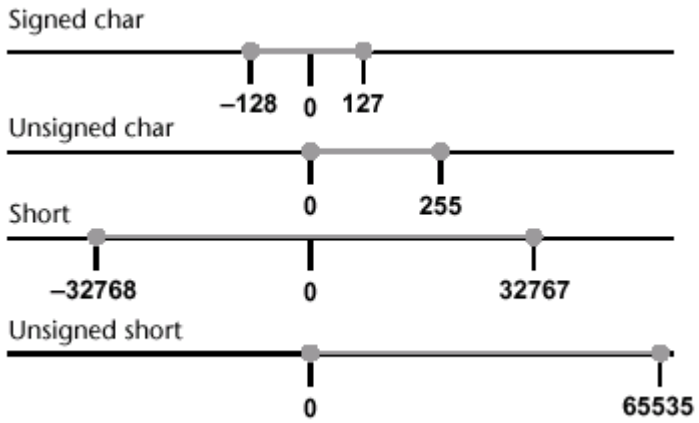
Integer Security

```
short s1 = 32000, s2 = 1500;
s1 = s1 + s2;
printf("%h, %d\n", s1, s1);
```

```
char cresult, c1, c2, c3;
c1 = 100;   c2 = 90;  c3= -120
cresult = c1 + c2 + c3;
printf("%c, %d, %c, %d\n", cresult, cresult, c3, c3);
```








```
1. unsigned int l = ULONG_MAX;
2. char c = -1;

3. if (c == l) {
4.   printf("Why is -1 = 4,294,967,295???\n");
5. }
```

Source: https://www.codeguru.com/cpp/sample_chapter/article.php/c11111/Integer-Security.htm

Examples of SW Bugs

Null pointer dereference!

```
int *ptr = NULL;
printf("Value of ptr: %d\n", ptr);


int *p = 0;
*p = 1;


int length;
char *buff;

scanf ("%d", &length);
buff = (char *) malloc(length+1);     // always Not NULL?
strcpy(buff, "Hello");
```

```
void Pointer(int *ptr) {
    *ptr = *ptr + 5;
}

main(void) {
    int num = 10;

    Pointer(&num);
    Pointer(NULL);
}
```

Examples of SW Bugs

Operator precedence Logic Error

```
node *find(node **curr, val) {
  while (*curr != NULL)
    if (*curr->val == val) return *curr;
  else
    *curr = *curr->next;
}
```

The arrow operator -> and the dot operator . bind more tightly than dereference *, parenthesis would solve the problem.

```
int x, a, b, c, d, e, f;

a =7; b = 6; c=5; d=4; e=3; f = 2;
x = a & b + c * d && e ^ f == 7;
printf("x = %d\n", x);
```

Bad Software

Bad software is everywhere!

- NASA's Mars Lander (cost $125 million)
  - Mars Climate Obiter
  - Error in converting English and metric units of measure
- Denver airport (in 1994)
  - Buggy baggage handling system
  - Delayed airport opening by 11 months
  - Cost of delay exceeded $1 million/day
- MV-22 Osprey: Advanced military aircraft
  - Lives have been lost due to faulty software

오스프리는 벨사와 보잉에 의해 개발됐으며 2000년 두 차례 실험비행에서 기술적 결함으로 23명의 사망자를 낸 후 오스프리 프로그램은 거의 폐지됐었지만 이후 이라크와 아프가니스탄에 배치돼 있으며, 일부는 네팔 지진피해 복구 작업을 지원하고 있다.

SW bugs can prove deadly

Bad software is everywhere!

- **Gamers May Fight Deadly Software Bugs in US Military Weapons [Jan. 23 2012, Technews]**
  - A lesson learned when a buggy Patriot missile defense system failed to intercept a Scud missile that killed 28 American soldiers during the first Gulf War in 1991.
  - To prevent such weapons disasters, the U.S. military wants to transform dull bug-hunting tasks into fun problem-solving games that attract swarms of online players

SW bugs can prove deadly

- **Bug can cause deadly failures when anesthesia device is connected to cell phones [Apr. 23 2014, ars-technica]**
  - Federal safety officials have issued an urgent warning about software defects in an anesthesia delivery system that can cause life-threatening failures at unexpected times, including when a cellphone or other device is plugged into one of its USB ports.
  - Spacelabs Healthcare is recalling the ARKON Anesthesia System with Version 2.0 Software due to a software defect.

의료기관이 사용하는 마취기는 보통 마취기계 본체 외에도 인공호흡기와 모니터 등이 세트로 이뤄져 있다. 그런데 마취기 소프트웨어에 취약점이 발견됐다고 한다.

문제가 된 마취기 모니터는 태블릿 같은 형태로 생겼고 USB 포트가 있다. 여기에 USB 케이블을 이용해 스마트폰 같은 기기를 마취기에 연결하게 되면 환자의 생명에 위협을 가할 수 있는 사태가 발생한다는 것이다.

미 식품의약안전국 FDA 역시 의료기간에 리콜에 따라 경고를 하고 있다. 이에 따르면 버그로 인해 마취기가 멈출 수 있으며 마취기에 붙어 있는 USB 포트에 디지털기기를 충전하면 작동이 중지된다는 것이다. 이에 따라 사망을 초래할 수 있는 심각한 결함이지만 다행스럽게도 아직까지 사망자는 보고되고 있지 않다.

anesthesia: (의학) 마취, 무감각증,

Summary

- Software is ubiquitous
- There are so many buggy programs

- Examples Software Bugs/Flaws
  - Improper initialization
  - Side effects
  - Scoping
  - Control flows
  - Integer security
  - Null pointer dereference
  - Operator precedence logic error

- Bad bugs can prove deadly