山东大学 计算机科学与技术 学院

云计算技术 课程实验报告

学号：202200130048	姓名： 陈静雯	班级： 6
实验题目：MD5 Hash 函数		
实验学时：2	实验日期： 2025.5.14	

实验目的：通过编程了解 MD5 算法，加深对 Hash 函数的认识，加深对 MD5 碰撞及其原理的理解。

具体内容：

1）使用 md5collgen 生成两个 MD5 值相同的文件。

2）生成两个 MD5 值相同但输出不同的两个可执行文件。

3）通过上面的实验，请解释为什么可以做到不同行为的两个文件具有相同的 MD5 值？

硬件环境：

计算机一台

软件环境：

Linux 或 Windows

实验步骤：

1）使用 md5collgen 生成两个 MD5 值相同的文件。

2）生成两个 MD5 值相同但输出不同的两个可执行文件。

3）通过上面的实验，请解释为什么可以做到不同行为的两个文件具有相同的 MD5 值？

实验前置知识：

一、基础知识

1. MD5 哈希算法
   - 定义：MD5 是一种广泛使用的密码哈希函数，可将任意长度数据映射为 128 位（16 字节）的哈希值。
   - 特性：
     - 确定性：相同输入始终得到相同哈希。
     - 抗碰撞性（已破解）：理论应难以找到两个不同输入具有相同哈希，但 MD5 已存在高效碰撞攻击方法（如 Wang 攻击）。
     - 雪崩效应：输入微小变化导致哈希值巨大差异。
2. 碰撞攻击（Collision Attack）
   - 定义：找到两个不同输入（M 和 M'），使得 MD5(M) = MD5(M')。
   - 原理：利用哈希算法数学结构的漏洞，构造特定差异的输入块，使中间状态抵消差异，最终哈希一致。
3. 可执行文件结构（ELF 格式）
   - 段（Section）：代码段（.text）、数据段（.data）、只读数据段（.rodata）等。
   - 文件偏移：段在文件中的物理位置（如.data 节偏移 0x3000）。
   - 全局变量存储：已初始化的全局变量通常位于.data 节。
4. 关键工具

- md5collgen：生成 MD5 碰撞块的工具。
- readelf/objdump：查看可执行文件段信息。
- dd：按字节分割/合并文件。

二、实验原理

1. 构造 MD5 碰撞的核心思想
   - 碰撞块（Collision Block）：生成两个不同的 128 字节块（B 和 B'），使得：MD5(前缀 + B) = MD5(前缀 + B')
   - 中间状态抵消：通过精心设计 B 和 B'的差异，使它们在 MD5 的压缩函数处理中产生相同的中间哈希值。
2. 可执行文件的 MD5 碰撞
   - 目标：生成两个可执行文件，满足：
     - 代码相同：程序逻辑一致。
     - 数据段不同：在.data 节插入碰撞块差异。
     - MD5 相同：整体文件哈希一致。
     - 行为不同：程序读取数据段差异，触发不同分支。

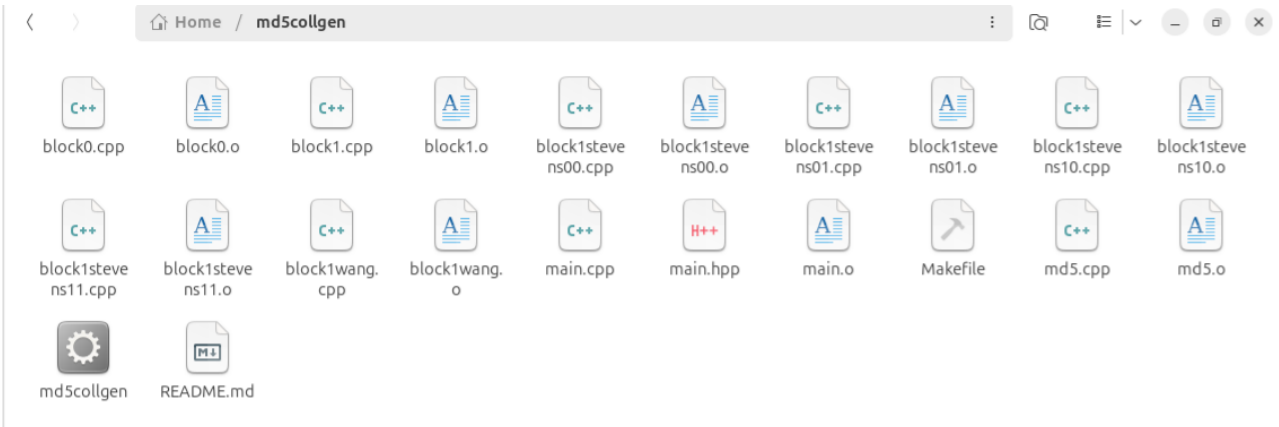
实验内容：

1. 安装 md5collgen

（1）安装依赖项

```
orange@orange-VMware-Virtual-Platform:~$ sudo apt-get install build-essential git libssl-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.43.0-1ubuntu7.2).
git set to manually installed.
The following additional packages will be installed:
  bzip2 dpkg-dev fakeroot libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libdpkg-perl libfakeroot libfile-fcntllock-perl
  lto-disabled-list
Suggested packages:
  bzip2-doc debian-keyring bzr libssl-doc
The following NEW packages will be installed:
  build-essential bzip2 dpkg-dev fakeroot libalgorithm-diff-perl
  libalgorithm-diff-xs-perl libalgorithm-merge-perl libdpkg-perl libfakeroot
  libfile-fcntllock-perl libssl-dev lto-disabled-list
```

（2）下载 md5collgen 源码



（3）编译

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ make
g++  -Wall -O  -c -o block0.o block0.cpp
g++  -Wall -O  -c -o block1.o block1.cpp
g++  -Wall -O  -c -o block1stevens00.o block1stevens00.cpp
g++  -Wall -O  -c -o block1stevens01.o block1stevens01.cpp
g++  -Wall -O  -c -o block1stevens10.o block1stevens10.cpp
g++  -Wall -O  -c -o block1stevens11.o block1stevens11.cpp
g++  -Wall -O  -c -o block1wang.o block1wang.cpp
g++  -Wall -O  -c -o main.o main.cpp
g++  -Wall -O  -c -o md5.o md5.cpp
g++ -o md5collgen block0.o block1.o block1stevens00.o block1stevens01.o block1st
evens10.o block1stevens11.o block1wang.o main.o md5.o
orange@orange-VMware-Virtual-Platform:~/md5collgen$ sudo cp md5collgen /usr/loca
l/bin/
```

2. 生成两个 MD5 值相同的文件

（1）新建 prefix.txt

```
This is a prefix message.
```

（2）生成碰撞文件

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ md5collgen -p prefix.txt -o out1.bin out2.bin
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1.bin' and 'out2.bin'
Using prefixfile: 'prefix.txt'
Using initial value: 6fa453be7f8370ba64cd5870d57683e4

Generating first block: ..
Generating second block: S00...
Running time: 0.253526 s
```

（3）验证 MD5 值

```
Running time: 0.253526 s
orange@orange-VMware-Virtual-Platform:~/md5collgen$ md5sum out1.bin out2.bin
303fc99f9ae959a840bf7b669504647f  out1.bin
303fc99f9ae959a840bf7b669504647f  out2.bin
orange@orange-VMware-Virtual-Platform:~/md5collgen$
```

3. 生成两个 MD5 相同但输出不同的可执行文件

（1）编写 C 程序

```c
#include <stdio.h>
// 预留128字节的碰撞块空间
unsigned char data[128] = {0};

int main() {
    if (data[0] == 0x12) {  // 假设碰撞块影响data[0]
        printf("Output A\n");
    } else {
        printf("Output B\n");
    }
    return 0;
}
```

（2）编译

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ gcc collide.c -o collide
```

（3）定位数据段偏移

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ readelf -S collide | grep .data
  [18] .rodata           PROGBITS         0000000000002000  00002000
  [25] .data             PROGBITS         0000000000004000  00003000
```

.data 偏移为 3000

（4）分割可执行文件

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ dd if=collide of=prefix bs=1 count=$((0x3000))
12288+0 records in
12288+0 records out
12288 bytes (12 kB, 12 KiB) copied, 0.0333593 s, 368 kB/s
```

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ dd if=collide of=suffix bs=1 skip=$((0x3000 + 128))
3568+0 records in
3568+0 records out
3568 bytes (3.6 kB, 3.5 KiB) copied, 0.0070509 s, 506 kB/s
```

（5）生成碰撞块

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ md5collgen -p prefix -o out1 out2
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'out1' and 'out2'
Using prefixfile: 'prefix'
Using initial value: 7da8c73c962fde2a0edb4d14a89ed5aa

Generating first block: ............
Generating second block: S10.....................................................
Running time: 7.613234 s
```

（6）合并文件生成可执行文件，将碰撞块与后缀合并

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ cat out1 suffix > collide1
orange@orange-VMware-Virtual-Platform:~/md5collgen$ cat out2 suffix > collide2
orange@orange-VMware-Virtual-Platform:~/md5collgen$ chmod +x collide1 collide2
```

（7）验证并运行

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ ./collide1
Output A
```

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ ./collide2
Output B
```

```
orange@orange-VMware-Virtual-Platform:~/md5collgen$ md5sum collide1 collide2
f0b451b0e23bb1f45c5ec258b866b806  collide1
f0b451b0e23bb1f45c5ec258b866b806  collide2
```

可见 md5 相同但是输出不同。

结论分析与体会：

为什么可以做到不同行为的两个文件具有相同的 MD5 值？

1. MD5 碰撞漏洞：md5collgen 生成的 out1 和 out2 在附加相同后缀后，MD5 哈希值相同。
2. 程序行为差异：碰撞块被写入预留的 collision_block 数组，程序通过读取该数组触发不同分支。
3. 文件结构完整性：前缀和后缀保持可执行文件格式（如 ELF 头）完整，确保文件可执行。