制造MD5算法的散列值碰撞 实验报告

计97 朱美霖 2019013294

实验目的

本实验通过`fastcoll`程序生成两个具有相同`MD5`散列值的文件，借此体会散列函数对文件校验的工作原理，并进一步加深对散列值碰撞的理解，进而加深对数据安全问题的认识。

实验内容

环境设置

- 操作系统：`Windows 11 Pro 21H2`
- `fastcoll`工具源自官网下载`Win32`可执行文件
- 可执行文件`test.exe`由`test.c`编译得到，为最简单的`C`语言`Hello world`程序：

```c
#include <stdio.h>

int main() {
    printf("Hello world");
    getchar();
    return 0;
}
```

实验步骤

具体的实验步骤依照书中的提示即可，如下图所示：

```
PS C:\Users\meilinzhu\Desktop\Cyberspace Safety\Experiments\制造MD5算法的散列值碰撞> .\fastcoll_v1.0.0.5.exe -p .\test.exe -o m1.exe
 m2.exe
MD5 collision generator v1.5
by Marc Stevens (http://www.win.tue.nl/hashclash/)

Using output filenames: 'm1.exe' and 'm2.exe'
Using prefixfile: '.\test.exe'
Using initial value: b5e20da33ea4a1039c5d4ffab1b78bcc

Generating first block: ..........
Generating second block: W....
Running time: 3.774 s
PS C:\Users\meilinzhu\Desktop\Cyberspace Safety\Experiments\制造MD5算法的散列值碰撞> certutil -hashfile .\m1.exe MD5
MD5 hash of .\m1.exe:
bb13cb7a4d66fdd048f5b9982342afe9
CertUtil: -hashfile command completed successfully.
PS C:\Users\meilinzhu\Desktop\Cyberspace Safety\Experiments\制造MD5算法的散列值碰撞> certutil -hashfile .\m2.exe MD5
MD5 hash of .\m2.exe:
bb13cb7a4d66fdd048f5b9982342afe9
CertUtil: -hashfile command completed successfully.
PS C:\Users\meilinzhu\Desktop\Cyberspace Safety\Experiments\制造MD5算法的散列值碰撞> certutil -hashfile .\m1.exe SHA1
SHA1 hash of .\m1.exe:
0301fe3948f6ab17f4faea625facf1ba8e7a222a
CertUtil: -hashfile command completed successfully.
PS C:\Users\meilinzhu\Desktop\Cyberspace Safety\Experiments\制造MD5算法的散列值碰撞> certutil -hashfile .\m2.exe SHA1
SHA1 hash of .\m2.exe:
9d53a5e027edc581934159a92f537ec438ea4046
CertUtil: -hashfile command completed successfully.
```

实验结果

可以发现 `fastcoll` 将 `test.exe` 作为蓝本生成了 `m{1,2}.exe` 两个新的可执行文件，它们具有相同的 `MD5` 散列值 `bb13cb7a4d66fdd048f5b9982342afe9`，但 `SHA1` 散列值不同。

`fastcoll` 论文中提到算法的复杂度为 $2^{32.3}$，实际运行只需要花费 `3.774` 秒（该例子）就可以生成具有相同 `MD5` 散列值的两个文件。

根据本次实验，我们可以得到，由于已经可以在可接受的时间范围内完成关于 `MD5` 算法的第二原像攻击，仅凭 `MD5` 算法进行基于散列加密的文件校验已经不再可靠。