MD5 Collision Attack Lab

Lab Report

Integrated Master in Informatics and Computing Engineering

Security in Computer Systems

Group 4:
João Fidalgo - 201303098
João Loureiro - 200806067
Ka Chon Ho - 201711244
Paulo Costa - 201206045

Faculdade de Engenharia da Universidade do Porto
Rua Roberto Frias, sn, 4200-465 Porto, Portugal

March 31, 2018

1 Introduction [1]

A secure one-way hash function needs to satisfy two properties: the one-way property and the collision-resistance property. The one-way property ensures that given a hash value h, it is computationally infeasible to find an input M, such that $hash(M) = h$. The collision-resistance property ensures that it is computationally infeasible to find two different inputs $M1$ and $M2$, such that $hash(M1) = hash(M2)$.

Several widely-used one-way hash functions have trouble maintaining the collision-resistance property. At the rump session of CRYPTO 2004, Xiaoyun Wang and co-authors demonstrated a collision attack against MD5 [2]. In February 2017, CWI Amsterdam and Google Research announced the SHAttered attack, which breaks the collision-resistance property of SHA-1 [3]. While many students do not have trouble understanding the importance of the one-way property, they cannot easily grasp why the collision-resistance property is necessary, and what impact these attacks can cause.

The learning objective of this lab is for students to really understand the impact of collision attacks, and see in first hand what damages can be caused if a widely-used one-way hash function's collision-resistance property is broken. To achieve this goal, students need to launch actual collision attacks against the MD5 hash function. Using the attacks, students should be able to create two different programs that share the same MD5 hash but have completely different behaviors. This lab covers a number of topics described in the following:

- One-way hash function
- The collision-resistance property
- Collision attacks
- MD5

2 Setup

The lab uses a tool called "Fast MD5 Collision Generation", which was written by Marc Stevens; the name of the binary is called *md5collgen* in our demonstration.

The result of the work of Marc Stevens et al as well as the source code of the *md5collgen* can be found at https://www.win.tue.nl/hashclash

To setup our machine we will need to download the source code for the *md5collgen*, install the dependencies and compile the code. The following steps are for a Debian like system such as Ubuntu.

Download the source code with the following command:

```
$ wget https://www.win.tue.nl/hashclash/fastcoll_v1.0.0.5-1_source.zip
--2018-03-28 22:00:54--  https://www.win.tue.nl/hashclash/fastcoll_v1.0.0.5-1_so
urce.zip
Resolving www.win.tue.nl (www.win.tue.nl)... 131.155.11.13
Connecting to www.win.tue.nl (www.win.tue.nl)|131.155.11.13|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 27567 (27K) [application/zip]
Saving to: `fastcoll_v1.0.0.5-1_source.zip'

100%[======================================>] 27,567      170K/s   in 0.2s

2018-03-28 22:01:02 (170 KB/s) - `fastcoll_v1.0.0.5-1_source.zip' saved [27567/2
7567]

$
```

Then we need to extract the contents of the zip file we just downloaded.

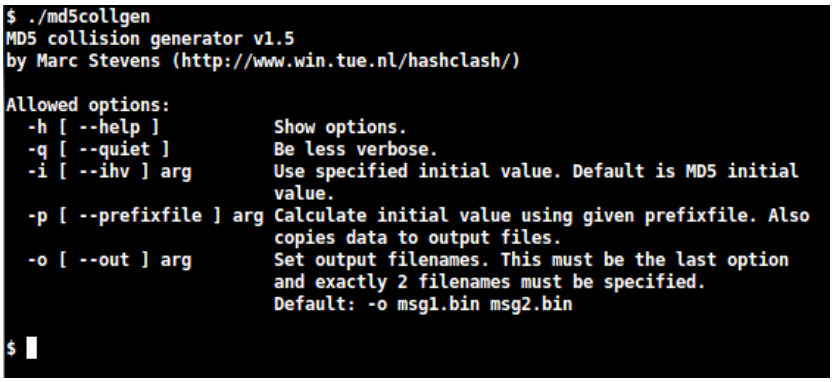
```
$ unzip fastcoll_v1.0.0.5-1_source.zip
Archive:  fastcoll_v1.0.0.5-1_source.zip
  inflating: block0.cpp
  inflating: block1.cpp
  inflating: block1stevens00.cpp
  inflating: block1stevens01.cpp
  inflating: block1stevens10.cpp
  inflating: block1stevens11.cpp
  inflating: block1wang.cpp
  inflating: main.cpp
  inflating: main.hpp
  inflating: md5.cpp
$
```

Before we can compile the code we have extracted we need to install three boost libraries: *system*, *filesystem* and *program-options*.

```
$ sudo apt-get install libboost-system-dev libboost-filesystem-dev libboost-prog
ram-options-dev
[sudo] password for seed:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  language-pack-kde-en language-pack-kde-en-base kde-l10n-engb
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  libboost-filesystem1.46-dev libboost-filesystem1.46.1
  libboost-program-options1.46-dev libboost-program-options1.46.1
  libboost-system1.46-dev libboost-system1.46.1 libboost1.46-dev
Suggested packages:
  libboost1.46-doc libboost-date-time1.46-dev libboost-graph1.46-dev
  libboost-iostreams1.46-dev libboost-math1.46-dev libboost-python1.46-dev
  libboost-random1.46-dev libboost-regex1.46-dev
  libboost-serialization1.46-dev libboost-signals1.46-dev
  libboost-test1.46-dev libboost-thread1.46-dev libboost-wave1.46-dev xsltproc
  doxygen default-jdk fop
The following NEW packages will be installed:
  libboost-filesystem-dev libboost-filesystem1.46-dev
  libboost-filesystem1.46.1 libboost-program-options-dev
  libboost-program-options1.46-dev libboost-program-options1.46.1
```

For the final step of our setup we need to compile the *md5collgen*.

```
$ g++ block* main.cpp md5.cpp -lboost_system -lboost_filesystem -lboost_program_
options -Wall -o md5collgen
$
```



3 Lab Tasks

3.1 Task 1: Generating Two Different Files with the Same MD5 Hash

3.2 Task 2: Understanding MD5's Property

3.3 Task 3: Generating Two Executable Files with the Same MD5 Hash

3.4 Task 4: Making the Two Programs Behave Differently

References

[1] Syracuse University Wenliang Du. *SEED Labs – MD5 Collision Attack Lab*. URL: http://www.cis.syr.edu/~wedu/seed/Labs_16.04/Crypto/Crypto_MD5_Collision/Crypto_MD5_Collision.pdf. (accessed: 28.03.2018).

[2] Xiaoyun Wang et al. "Collisions for Hash Functions MD4, MD5, HAVAL-128 and RIPEMD." In: 2004 (Jan. 2004), p. 199.

[3] Marc Stevens et al. "The first collision for full SHA-1". In: *Annual International Cryptology Conference*. Springer. 2017, pp. 570–596.