Nat McHugh

An important site on a minor internet.

Wednesday, May 6, 2015

How to make two binaries with the same MD5 hash

One question I was asked when I demo'd creating two PHP files with the same hash is; does it work on compiled binaries?

Well the answer is yes in fact that is where I first got the idea from, in this demo.

That example uses a C program as both the target and also to do the binary manipulation, which slightly of obscures the way it works. It also makes use of an old very slow implementation on the Wang attack to generate the collisions. To better and more quickly show how it works for an upcoming talk I have created a really simple example using a PHP script to manipulate a binary once compiled.

I have put all the code used here on github.

Below is the super simple C program it compares two strings and if they don't match it prints out an ASCII art picture of an angel. If they do match you get a picture of a devil.

```
1
2  #include <string.h>
3  #include <stdio.h>
4
5
6  #define DUMMY "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" \
7   "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" \
8   "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" \
9   "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" \
10  "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA" \
11  "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
12
13 int angel();
14 int devil();
15
16 char *dummya = DUMMY "A";
17 char *dummyb = DUMMY "B";
18
19 int main() {
20   if (strcmp(dummya, dummyb) != 0) {
21     return angel();
22   } else {
23     return devil();
24   }
25 }
26
27
28 int angel() {
29     fprintf(stdout, ".                                    ,\n");
30     fprintf(stdout, ")).              -====-              ,((\n");
31     fprintf(stdout, ")))).                                ,(((\n");
32     fprintf(stdout, "))))).           .::::.           ,(((((\n");
33     fprintf(stdout, "))))))))).       :. .:        ,(((((((\n");
34     fprintf(stdout, "`))))))))))).    : - :    ,((((((((((\n");
35     fprintf(stdout, "))))))))))))))_:' ':_((((((((((((('\n");
```

Twitter

```
36      fprintf(stdout, " `)))))))))))).-' \\___/ '-._((((((((((\n");
37      fprintf(stdout, "  `))))_._.-' __)(     )(_  '-._._((((\n");
38      fprintf(stdout, "   `))'---)___)))'\\_ _/'((((__(---'(('\n");
39      fprintf(stdout, "     `))))))))))))|' '|((((((((((('\n");
40      fprintf(stdout, "      `)))))))))/'   '\\((((((((((\n");
41      fprintf(stdout, "         `)))))))|     |(((((((\n");
42      fprintf(stdout, "          `))))))|     |((((((\n");
43      fprintf(stdout, "                /'     '\\\n");
44      fprintf(stdout, "               /'       '\\\n");
45      fprintf(stdout, "              /'         '\\\n");
46      fprintf(stdout, "             /'           '\\\n");
47      fprintf(stdout, "             '---..___..---'\\\n");
48    return 0;
49  }
50
51  int devil() {
52      fprintf(stdout, "        _.---**""**-.\n");
53      fprintf(stdout, "._   .-'           /|`.\n");
54      fprintf(stdout, " \\`.'             / |  `.\n");
55      fprintf(stdout, "  V              (  ;    \\\n");
56      fprintf(stdout, "  L       _.-  -. `'      \\\n");
57      fprintf(stdout, " / `-. _.'       \\         ;\n");
58      fprintf(stdout, ":            __   ;    _   |\n");
59      fprintf(stdout, ":`-.___.+-*\"': `  ;  .' `. |\n");
60      fprintf(stdout, " |`-/     `--*'   /  /  /`.\\|\n");
61      fprintf(stdout, ": :              \\    :`.| ;\n");
62      fprintf(stdout, "| |   .           ;/ .' ' /\n");
63      fprintf(stdout, ": :  / `            :__.'\n");
64      fprintf(stdout, " \\`._.-'       /     |\n");
65      fprintf(stdout, "  : )         :      ;\n");
66      fprintf(stdout, "  :----.._    |     /\n");
67      fprintf(stdout, " : .-.    `.       /\n");
68      fprintf(stdout, "  \\     `._       /\n");
69      fprintf(stdout, "  /`-            /\n");
70      fprintf(stdout, " :             .'\n");
71      fprintf(stdout, "  \\ )       .-'\n");
72      fprintf(stdout, "   `-----*\"'\n");
73    return 0;
74  }
```

demo.c hosted with ♡ by GitHub view raw

It can be compiled with gcc and executed simply by doing

```
longEgg$ gcc -o demo ./demo.c
longEgg$ chmod a+x demo
longEgg$ ./demo
```

Executing the program will print out the angel since the two strings differ in the last letter.

Now we have our compiled binary we need to do a bit of mucking about with it. What we are going to do is insert a MD5 collision into the long string of A's of the dummy text. We only need to insert two blocks of 64 bytes but we need to insert it at the beginning of a block i.e. when the byte length is a multiple of 64 bytes.

```
<?php

include __DIR__.'/MD5.php';
$inFile = __DIR__.'/demo';
$dummyText = str_pad('', 64, 'A');

function replaceDummyText($input, $replacment, $position)
{
    return substr_replace($input, $replacment, $position, strlen($replacment
}

```

Tweets by @natmchugh

natmchugh @natmchugh

@katie_fenn ok 🔥🔥🔥

17 May

natmchugh @natmchugh

@katie_fenn I have one if you need to borrow one.

17 May

natmchugh @natmchugh

Ah yes short Weierstrass form versus Edwards form, very amusing.

12 May

natmchugh @natmchugh

@katie_fenn Sadly not available beta.companieshouse.gov.uk/search/company...+

Embed View on Twitter

Popular Posts

Create your own MD5 collisions

A while ago a lot of people visited my site (~ 90,000) with a post about how easy it is to make two images with same MD5 by using a chos...

How I created two images with the same MD5 hash

I posted the following images the other day which although looking totally different have exactly the same MD5 hash (e06723d4961a0a3f950e...

How I made two PHP files with the same MD5 hash

I recently posted a link on twitter to two PHP scripts which have different behaviours but the same MD5 hash. To verify this download the fi...

How to make two binaries with the same MD5 hash

One question I was asked when I demo'd creating two PHP files with the same hash is; does it work on compiled binaries? Well the answ...

Three way MD5 collision

Previously I explained how I created two images one of James Brown the other of Barry White with the same MD5 hash. At the end of the post I...

The Magic Words are Squeamish Ossifrage - factoring RSA-129 using CADO-NFS

This thing got long and can basically be summarised as: I factored smaller RSA numbers on free cloud computing using excellent open sour...

Hash Collisions Reading List

```php
function findDummyText($filestring, $dummyText)
{
    $pos = 0;
    $chunks = str_split($filestring, 64);
    foreach ($chunks as $chunk) {
        if ($chunk == $dummyText)  {
            break 1;
        }
        $pos++;
    }
    return $pos*64;
}

// read in the original binary file in
$filestring = file_get_contents($inFile);

// find the place where we have the dummy string and its at start of a 64 by
$pos = findDummyText($filestring, $dummyText);
printf('I want to replace %d bytes at position %d in %s'.PHP_EOL, 128, $pos,
$firstPart = substr($filestring, 0, $pos);

//find the IV up to the point we want to insert then print that out
$iv = md5_hash($firstPart);
printf('Chaining variable up to that point is %s'.PHP_EOL, $iv);

if (!file_exists(__DIR__.'/a')) {
    print('Run fastcoll to generate a 2 block collision in MD5'.PHP_EOL);
    return;
}

// replace the dummy text at the correct location
$good = replaceDummyText($filestring, file_get_contents(__DIR__.'/a'), $pos)
$bad  = replaceDummyText($filestring, file_get_contents(__DIR__.'/b'), $pos)

// find the secod dummy string
$secondDummyTextStart = strpos($good, str_pad('', 191, 'A'));

// serach back from where we inserted the collision first time so we can gra
// 192 bytes and use it to replace the second string
while ('A' == substr($filestring, $pos-1, 1)) {
    --$pos;
}

//the 192 butes of str1
$replacement = substr($good, $pos, 192);

// replace str1 with 192 bytes cut from of the files
// the file it came from will then compare str1 and str2 to 0
$good = replaceDummyText($good, $replacement, $secondDummyTextStart);
file_put_contents(__DIR__.'/devil', $good);
printf('Just output new file %s with hash %s'.PHP_EOL, __DIR__.'/devil', md5

$bad = replaceDummyText($bad, $replacement, $secondDummyTextStart);
file_put_contents(__DIR__.'/angel', $bad);
printf('Just output new file %s with hash %s'.PHP_EOL, __DIR__.'/angel', md5
```

long_egg.php hosted with ♡ by GitHub view raw

When we run the php script over it the first time it finds such a location and calculates the value of the four chaining variables of MD5 at that point in the file. It prints out the hex values concatenated together as a hash. We can now take that value and search for an MD5 collision with that initial state. The best MD5 collision finder is Marc Stevens fastcoll. It can typically find collisions in a couple of seconds using a a variant of the Wang attack. After downloading it you will need to compile it. There should be a Makefile for it in the code on

Lately in an effort to code up and properly understand the Wang attack on the MD4 family of hash functions I've been reading a lot of pa...

Images with colliding MD5 hash

These images of James Brown and Barry White have the same MD5 hash e06723d4961a0a3f950e7786f3766338. Don't believe me. You can...

Bug Bounty - Remote Code Execution in Magento 1.x

Magento is a popular ecommerce solution written in PHP. It is widely used for web shops both large and small. The most current product is Ma...

MD5 collisions in ssh keys

You can insert MD5 collision blocks in many data formats and if you do it right the result will be 2 objects that differ but which when pass...

Labels

- Cloud
- Hash functions
- Maths
- MD5
- PHP
- RSA

Blog Archive

- ▶ 2017 (1)
- ▼ 2015 (8)
  - ▶ September (1)
  - ▶ July (2)
  - ▶ June (1)
  - ▼ May (1)
    - How to make two binaries with the same MD5 hash
  - ▶ March (1)
  - ▶ February (1)
  - ▶ January (1)
- ▶ 2014 (5)

Links

- Inviqa
- Sheffield PHP User Group
- Fishtrap

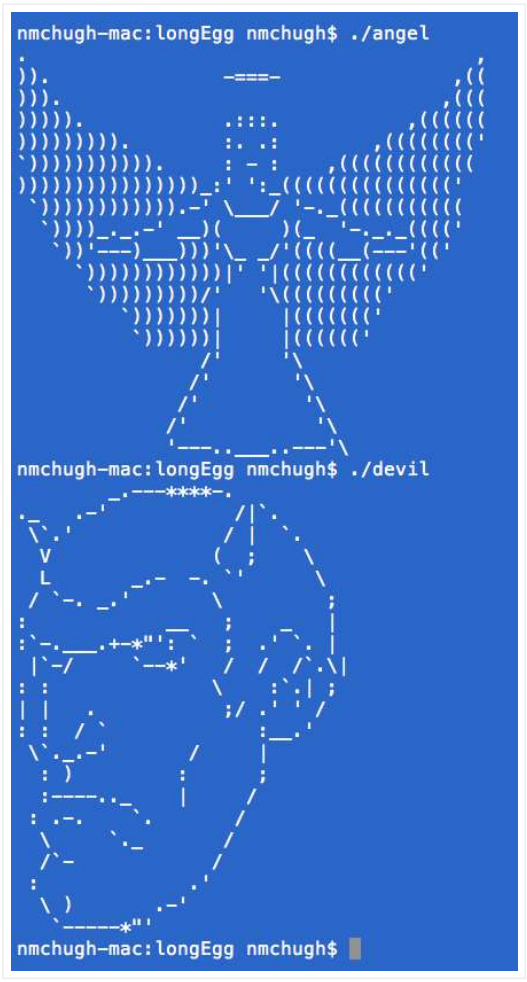
github. Running it specifying the initial state and output files is shown below.

```
longEgg$ wget https://www.win.tue.nl/hashclash/fastcoll_v1.0.0.5-1_s
longEgg$ unzip fastcoll_v1.0.0.5-1_source.zip
longEgg$ make
longEgg$ chmod a+x fastcoll
longEgg$./fastcoll -i c15cfe39c40e47f5b8ae31e6658fd1bd -o a b
```

The -o option specifies the output files and so will create two new files a and b which contain 2 blocks of binary data. These blocks only work as MD5 collisions within the binary at that point. Running the php script for a second time will create two copies of the original compiled binary with the collisions inserted in the appropriate places.

```
longEgg$ I want to replace 128 bytes at position 6528 in colliding_b
longEgg$ Chaining variable up to that point is c15cfe39c40e47f5b8ae31
longEgg$ Just output new file /Users/nmchugh/longEgg/devil with hash
longEgg$ Just output new file /Users/nmchugh/longEgg/angel with hash
```

So now we have created two more files angel and devil. Running each of those should give different outputs.



But they should have the same MD5 value.

```
longEgg$ md5 angel devil
MD5 (angel) = dea9dc288b6c56626997ce86ca8eb6da
MD5 (devil) = dea9dc288b6c56626997ce86ca8eb6da
```

Posted by Nathaniel McHugh at 1:38 PM

G+1 +13 Recommend this on Google

Labels: Hash functions, MD5, PHP

1 comment:

Blake Bond Monday, March 06, 2017

This comment has been removed by a blog administrator.

Reply