Nat McHugh

An important site on a minor internet.

Friday, October 31, 2014

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 (e06723d4961a0a3f950e7786f3766338) .

Twitter

The images were just two I lifted from the web in fact I could have chosen any image or indeed any arbitrary data and created a collision with it.

Why is this surprising?

MD5 was designed as a cryptographic hash function. These are supposed to possess the following 3 properties as per wikipedia (http://en.wikipedia.org/wiki/Cryptographic_hash_function#Properties)

- *Pre-image resistance*

  Given a hash h it should be difficult to find any message m such that $h = hash(m)$.

- *Second pre-image resistance*

  Given an input m_1 it should be difficult to find another input m_2 such that $m_1 \neq m_2$ and $hash(m_1) = hash(m_2)$.

- *Collision resistance*

  It should be difficult to find two different messages m_1 and m_2 such that $hash(m_1) = hash(m_2)$. Such a pair is called a cryptographic hash collision.

The two images above clearly demonstrate that MD5 lacks the final property. MD5 is broken as a cryptographic hash function.

To search though all possible MD5 values is 2^{128} operations which is massive. To be in with a good chance of finding a collision would take ~ 2^{64} operations which is again far too big for normal computing.

How did I create the images?

This type of collision is has been termed a chosen prefix collision. In this case the image data is the prefix or to be more exact the internal state of the MD5 algorithm after processing the image is. You can't see the added binary data at the end of jpeg images as it is preceded with an End Of Image JPEG marker.

Chosen prefix collisions for MD5 were first successfully shown in 2007 in this paper http://www.win.tue.nl/hashclash/ChosenPrefixCollisions/ . The attack uses iterations of differential analysis of MD5. The first successful differential analysis was demonstrated by Xiaoyun Wang in her 2005 paper How to Break MD5 and Other Hash Functions.

In a previous post I used an improved version of this differential path to create two PHP files with the same MD5 hash. Where those files differed by only a few bits and the data before the collision had to be identical. In a chosen prefix collision the data preceding the specially crafted collision blocks can be completely different.

The chosen prefix collision attack works by repeatedly adding 'near collision' blocks which gradually work to eliminate the differences in the internal MD5 state until they are the same. Before this can be done the files must be of equal length and the bit differences must be of a particular form. This requires a brute force 'birthday' attack which tries random values until two are found that work, it does however have a much lower complexity than a complete brute force attack.

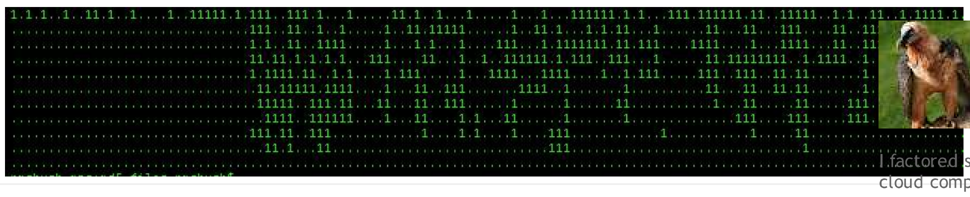
If the attack sounds complicated to do in practice fortunately Marc Stevens has created framework for automated finding of differential paths and using them to create chosen pre-fix collisions. It is available at https://code.google.com/p/hashclash/ . It is intended mainly as a research tool but there is a GUI and pre-built binaries for windows available. I chose to run it on linux, however, using a bash script which can automate the repetitive steps needed.

Here are the MD5 states following each successive block (these are unpadded versions of MD5 algorithm).

```
601034f03377d68d68a74f71b0d76bf4 c924b00ad433ccc979b8e79e6925f28e
0e5453c5c7deabc5e23331c415780ecf 0e5453c521968d3df17653c224bb30cd
8d43bcaea7f738cbbbae37b1a5ec4f3c 8d43bcae018f1a43cad159afb40f723a
ea12c8b8645e161d589c69dcf845dd60 ea12c8b8bef6f79467c08bda076aff5e
8ec47a1d73c2267dfb7686b83a3554fb 8ec47a1dce5608750a97a8b6495576f9
daa7508daa16fe67f09ede251abd83a2 daa7508dc5aadd5fffbefe2329dda3a0
c570b0b7b22e3a2545432f2a96baf83a c570b0b7cec2591d55634f28a6da1839
00d3111f51f505e81d5f5db537ed64a4 00d3111f6d0926e22d7f7db5470d85a4
73e0e4d069fdac380cfe2cf0e7ba47fb 73e0e4d0851dad321c1e2df0f7da47fb
583a08b75fa22174307a24df6109b9ec 583a08b76bc22174309a24df6129b9ec
e2bdd99bb0bcc66557192eec1f36ff44 e2bdd99bb0bcc66557192eec1f36ff44
```

The first line is the state after processing the original images where the two hashes are unrelated. The second line shows the state after padding to equal length and the addition of the 'birthday' bits. As you can see the first four bytes of the hash are the same. Each of the next nine lines shows the hashes gradually converging until they are there are no differences. The last hash is different from the one calculated on the whole image as that one includes padding.

The image below shows the bit differences between the above hashes a '.' indicates they are the same a '1' indicates a difference.



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/companì...+

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

I ran HashClash on an AWS GPU instance. I cannot say with any certainty that this is the most efficient or cheapest option but it seemed to work reasonably quickly. In particular the 'birthdaying' step took much less time than I had expected. It finished in roughly 1hr originally this step had an estimated complexity of 2^{49} compression function calls. In his article Marc Stevens gives an estimate of 2 days for creating a complete collision on a PS3 in 2007. I found that I was able to run the algorithm in about 10 hours on an AWS large GPU instance bringing it is at about $0.65 plus tax.

I faced a few problems with the code as published and had to make some changes to the bash script and some of the C++ code related to saving the collisions. I don't know if the windows binaries that are published work any better as this seems to be where effort has most recently been expended. If anyone wants too know the changes I made let me know.

So I guess the message to take away here is that MD5 is well and truly broken. Whilst the two images have not shown a break in the pre-image resistance or second pre-image resistance, I cannot think of a single case where the use of a broken cryptographic hash function is an appropriate choice.

It was a chosen prefix collision attack similar to this that was used to produce a counterfeit ssl certificate used to sign the Flame malware as Microsoft and pass itself off as a Windows update.

What Next?

Well once you have two colliding files you can easily create a third. You just take the output of the first files and a third with any data you want file and use it as the starting point for another chosen prefix collision. I may do this and would welcome image suggestions.

Marc Stevens has also published a tool for detecting files which have been processed in this way. Running this tool over either of the files can detect that collision blocks have been added.

Update

I successfully created a third collision.

Posted by Nathaniel McHugh at 5:20 PM

G+1 +49 Recommend this on Google

Labels: Hash functions
Location: Sheffield, South Yorkshire, UK

No comments:

Post a Comment

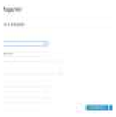
Enter your comment...

Comment as: Select profile... ▼

Publish Preview

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 e06723d4961a0a3f950e7786f 3766338. 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)
- ▼ 2014 (5)
  - ► November (2)
  - ▼ October (3)
    - How I created two images with the same MD5 hash
    - Images with colliding MD5 hash
    - How I made two PHP files with the same MD5 hash

Links

- Inviqa
- Sheffield PHP User Group
- Fishtrap