

Date: 2019-09-04 Author: Ian Rash Index: #3

We also finally got some passwords, but they were hashed, so we are going to need to figure out what method the manufacturer used. First thing to notice is that, in other captures, the PassWord field is always the same, meaning chances are we aren't dealing with a random [salt](#) every time. If they do use a salt it must be hard coded into the camera itself, but this is unlikely. What IS really weird about this hash is that it says it's "MD5" but the hash itself is only 8 bytes, where MD5 has 16. This means that there is some hashing protocol, but it is custom and built upon MD5.

I actually searched up and down and couldn't find anything about this hash format. I tried tools like [CyberChef](#) and a [hash calculator](#) to no success. Unfortunately, I couldn't find the hashing format used, so I asked the wonderful people on the [Voiding Warranties Discord](#), and admin [Retr0id](#) knew the hash type, Dahua! Since he helped me out, I'll plug his [very cool exploit](#) for a brand of wifi-enabled SD cards, go check it out, it's a super cool project!

In the end I used a bit of [source code](#) for the hashing system to write my own Dahua hashing for Crystal.

```

# Code translated from https://github.com/haicen/DahuaHashCreator/blob/master/DahuaHash.py
require "digest/md5"

module Dahua
  def self.compress(bytes : Slice(UInt8)) : Bytes
    i = 0
    j = 0
    output = Bytes.new(8, 0)
    while i < bytes.size
      output[j] = ((bytes[i].to_u32 + bytes[i+1].to_u32) % 62).to_u8

      if output[j] < 10
        output[j] += 48
      elsif output[j] < 36
        output[j] += 55
      else
        output[j] += 61
      end

      i = i+2
      j = j+1
    end
    output
  end

  def self.digest(password)
    md5_bytes = Digest::MD5.digest(password.encode("ascii"))
    compressed = compress(md5_bytes.to_slice)

    String.new(compressed)
  end
end

```

I'm not hashing expert, but this method to me screams, '[COLLISION](#)'. From my calculation, this system loses about 99.9% of the entropy in the hash, that's not even a joke, each character in the Dahua hash has a 62 possibilities, where the original MD5 hash has 65535 possibilities each (since the hashing algorithm takes two bytes from the MD5 hash for each one of it's characters). This means that for each MD5 hash, there is a total of $2^{(8*16)}$, which is a very big number, and reduces it down to 62^8 , which is a much much smaller number.

Using the digest method, we can attempt to determine our password hashes.

```
"" = tIJwpbo6
"password" = mF95aD4o
"abcdef" = vfMMASaj
"123456" = nTBCS19C
"asdfghjkl" = MajKjGGZ
"00000000000000000000000000000000" = IJ84MHIF
```

[Done] exited with code=0 in 0.586 seconds

TAGGED SKILLS:

 Reverse Engineering  Security Research  C