CSCI 400-02 Cryptography Lab
Topic: Hashes, Stashes, and Rainbows

Prof. Dietrich

September 18, 2017

Group size: 4 — Setup needed: Windows/MacOS X, Unix hosts, DETER (or closed network setup).

1 General description

This lab explores the dictionary attacks, as in running a dictionary (or list of words) against a cryptosystem to find a matching key. Choosing the correct dictionary and possibly the right modifications on each word can quickly yield a key, surpassing a simple brute-force attack.

2 File with samples and results

The files are in lab-crypto2.zip. It contains most files to get you started, including the samples and the challenges. In some cases, you still have to develop some scripts or write your own code to improve the results. The tools include John the Ripper, Cain & Able, and OphCrack.

3 Cracking passwords and hashes

In this part of the lab, we crack passwords, i.e. we apply dictionary attacks on the hashes. In most cases this assumes we are familiar with the cryptosystem or hash function used to create the ciphertext or hash, respectively. There will be one case where the hash function will be unknown. Possible targets: crypt (Unix), modified crypt (Blowfish), MD5, NTLM hashes, SHA1 hashes. Extensions are possible for ZIP files (or RAR, 7zip), as well as PDF and MS Office encryption.

3.1 Requirement

- You should use the tools provided in the file, but are not limited to them. If for some reason you choose not to, please justify your choice. If you wrote scripts or additional code, please include it.
- Your output should demonstrate that you have done the experiments, such as screenshots, Unix typescripts, or logs.

- Your output should be the passwords found, corresponding to their respective hash values or password entries. You should specify in each case how you cracked the password, e.g. by using a Swedish dictionary, applying the rule to substitute 3 for es, etc. As a last resort, a brute-force method may be used (some password crackers use that as a last mode/pass), but then you should specify that this was done.

4 Tasks

4.1 Finding Unix passwords

Using the tools of your choice, find the password that generated each of the following Unix crypt hash entries in the following entries from the `/etc/passwd` file:

- `root:x11UGsMj7vR.A`
- `michael:9rSp0Y.eGGDMc`
- `angelina:CZ5MOX8FdBWoQ`
- `john:NrrN.A7iPEVys`
- `neo:Qxw8mUWuWCUic`
- `cassandra:BHWyRvG9wuvHo`

4.2 Finding a preimage to an MD5 hash

Using the tools of your choice, find the probable origin text for the following MD5 hashes:

- `827ccb0eea8a706c4c34a16891f84e7b`
- `c9bc7129e72a5b85abba890fe63c30e3`
- `f33b165eb031df7dd412695bc3637f31`
- `05c73fd8d7e6db8d3dffd41ecb24a99b`
- `705913ae84d53241fa32e7bcfdf2b602`
- `ed62caf705899e50b3945b8ae0b74378`

4.3 Finding a preimage to a SHA-1 hash

Using the tools of your choice, find the probable origin text for the following SHA-1 hashes:

- `f97ba360c7cfdad2244c040ffa4b57d4224db970`
- `f184b549d3a4efb79fc69a331426680c57e637bc`
- `7b7a8d8e9435d1064967f8ba2a43eee1f7804f5e`
- `572cc530f965639297db129cc536a5cd5d0b7b9c`
- `1b1a52fb9b0791e0211c2af0d3155b80ac40405b`

4.4 Something a bit harder

4.4.1 Unlabeled hashes, single form

What hash function and text could have generated the following hashes? Note the length of the hashes and deduce which hash function could be a possible candidate.

- `f573b7dd8369faa58368bb167539106ff0e1e027`
- `0a4628c5ae0771b1d27ac71e2d2574dec4174f35fa0a6c5e7e0f98871fbcc657`
- `$2y$05$68LWs2pXiEvhVTacJ9PX/elfa1zq9jCBlv61hEw1pCvAHlaJyeMwW`
- `2e951c7436e3062065d2da6ac0462a844adcd4f8410589c3908c92397a4300c016ff`
  `3ac58244a9d5a39421a5894a794cbfc0e155b05ce9dfb805abe650ce9687` (yes, it's that long)

4.4.2 Keyed and nested hashes

Rather than straightforward hashes, these are variations thereof.

- A keyed MD5 hash (HMAC-MD5), with both key and text being proper names, key is a female name, text is an author, mixed alphabeticals, up to 6 characters: `8b5a3e95656026f9ce2f405e279adf06`

  For HMAC, we will assume the HMAC construction by Hugo Krawczyk, Mihir Bellare, and Ran Canetti (`RFC 2014`, February 1997), so that:

  ```
  Key = "Wizard"
  Text = "Toto, I've a feeling we're not in Kansas anymore"
  HMAC-MD5(key, text) = 32dd33edec6831a745ed575dfe623ddc
  ```

- A nested SHA2-256 hash: `3d91b58504a6cc3a159005ee7b16c7ae503ca6ac2a6a3c893837083c236b864a`

  We are assuming the construction to be such that:

  ```
  SHA2-256(SHA2-256(test)) = 7b3d979ca8330a94fa7e9e1b466d8b99e0bcdea1ec90596c0dcc8d7ef6b4300c
  ```

4.4.3 Something much harder: unusual hashes

These are hashes that differ from the normal setup. In this section, the hashes were generated using shell commands similar to these:

```
$ echo textgoeshere | openssl md5
bd88df09c73fadbb9b5c1e33ca452e4b
```

Note that this is a different result than using this command:

```
$ echo -n textgoeshere | openssl md5
96ad4fad21a0d24c732a00cf3450e2ae
```

That creates a challenge. The provided and otherwise available tools do not typically handle this type of construction, so you may have to modify existing tools to accommodate this setup, or develop your own, if you have not done so already for the earlier tasks.

Find the preimage to the following hashes, constructed as above:

- `cac6b6d7d43e5fc18b0272164215b8f1` (MD5)
- `2d202cb6a781edf6ac01d14e1dd6367497db90ba9e8ad721fd17239228a213f9` (SHA2-256)
- `76451d1128dc80db348a2db75a9d3c82242f679035639fca0902cba58fd64819fe62f`
  `41129e67e67076eec6882ab1eac5159c9305bf0c3075ff473a048471632` (SHA3-512)

4.5 Encrypted archive files

Archive files allow for encryption of the contents, which protect the files themselves, and sometimes even the file listing of the archive. Here we attempt to crack the encrypted archives.

4.5.1 ZIP

In your lab files, you will find a ZIP file called challenge400.zip. It contains some files inside, but the ZIP archive is protected by a password. The lab file contains sample instructions for cracking ZIP files.

For reference, the SHA-1 hash of the file challenge400.zip is:
SHA1(challenge400.zip)= a3b19f963c905c883657dbcad27c5a534d9b0e79

4.5.2 7ZIP

In your lab files, you will find a 7ZIP file called challenge400.7z. It contains some files inside, but the 7ZIP archive is protected by a password. Write a program or script to crack the password to get to the files. Include the program/script in your report.

Hints: the encryption is AES-256 and the password is something close to what you are reading. What are these files inside? For reference, the SHA-1 hash of the file challenge.7z is:
SHA1(challenge400.7z)= 73351ebb1445053bd76532a5f23c18e1a3bb389d

5 Word Problems

1. Summarize the attack techniques used by the tools.
2. Looking at the rainbow table approach, does it sound like a viable alternative?
3. How does JTR (or the others) improve on simple dictionary attacks?
4. Does using a GPU help? How so?
5. How would you improve these techniques further?

6 Deliverables

1. A report describing all your findings above.
2. A zip file containing:
   - Any data files found (e.g. from (7)ZIP files), texts found, including the particular technique used.
   - Answers to all word problems in Part 5.
3. Submit by class time on September 27, 2017.

7 Grading

Points will be subtracted if any of the pieces of the deliverables are missing or incomplete.