Uebung 4

Philip Magnus

January 8, 2025

HÜ4 - Passwords

0. Intro

The following documentation will show how the hashes given in excercise 4 were analyzed and eventually cracked.

0.1 Given hashes

The following list contains the given hashes for exercise 4:

```
5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8

ECprxlyOyUshcl60I9RBSQ==:3NfdMsGy2AAHhm27PTFe0A==

0x102a6bc65c8ec94b21725eb423d44149:dcd7dd32c1b2d80007866dbb3d315ed0

$2y$12$RjmAST8RecCMVv1DO4g9zOv3wQ4/vsHlWyC7FLBF/O7WL9SzVeA.m

GvkWX7r0M7mkLxQ743qaDQ==:0/xjqfLDZfAaeI5/s6zjlg==

0xf7c3bc1d808e04732adf679965ccc34ca7ae3441

$2y$10$NqnZ8O1xz0UdArMymrPjoOy2.WwLVqfFYxc1rkwwZdw9.f140Wq72

qoIP+jhQg+rY3SWAR+FdnZC/H0Gj86WoehIejMC71so=
(Salt: 1, N 2048, R 8, P 1, Length 32)

5ts7TXFWp/S2WqMEDXZXO/eEkIE=

64iGbek1bgraDo3VoxjOY4ftjKQj5yWujhG7UyDMCS4=
(Salt: 2, N 2048, R 8, P 1, Length 32)

$2y$10$fP0pAnFtq0aiOTfjLRGB1.rJU5RE1r/P1xcux8vHd6e1Zw2bJXe7y

Administrator:500:aad3b435b51404eeaad3b435b51404ee:957ed2442e3fece7bf7cfe1417115710:::
```

0.2 Installation of needed software

For the analysis of the given hashes the software hashid was installed. Hashid is a program wirtten in Python, which tries to identify hashes based on a regex ruleset.

```
pipx install hashid
```

For the installation on a Ubuntu sytem `pipx` was used. `pipx` will manage the Python venv and make `hashid` globally available.

After identifying the hash type `hashcat` was used to crack the hashes with a dictionary attack.

```
sudo apt install hashcat -y
```

`hashcat` was install through the Ubuntu repositories and `apt`.

1. Analysis

The following section will describe the process of analyzing the given hashes to determine their hash types.

1.1 Hashid

To try and determine the type of hash used the list of hashes was put into a `hashes.txt` file.

The `hashes.tyt` file was passed to `hashid`:

```
 hashid -e -m hashes.txt -o hashid.out
```

The `-e` flag is used to get an extended output with all possible hash types. `-m` is used to get the corresponding mode used by `hashcat` to try and crack the hash. The flag `-o` sets the outputfile for hashid.

The following is the output of `hashid`:

```
Analyzing '5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8'
[+] SHA-1 [Hashcat Mode: 100]
[+] Double SHA-1 [Hashcat Mode: 4500]
[+] RIPEMD-160 [Hashcat Mode: 6000]
[+] Haval-160
[+] Tiger-160
[+] HAS-160
[+] LinkedIn [Hashcat Mode: 190]
[+] Skein-256(160)
[+] Skein-512(160)
[+] MangosWeb Enhanced CMS
[+] sha1(sha1(sha1($pass))) [Hashcat Mode: 4600]
[+] sha1(md5($pass)) [Hashcat Mode: 4700]
[+] sha1($pass.$salt) [Hashcat Mode: 110]
[+] sha1($salt.$pass) [Hashcat Mode: 120]
[+] sha1(unicode($pass).$salt) [Hashcat Mode: 130]
[+] sha1($salt.unicode($pass)) [Hashcat Mode: 140]
[+] HMAC-SHA1 (key = $pass) [Hashcat Mode: 150]
[+] HMAC-SHA1 (key = $salt) [Hashcat Mode: 160]
[+] sha1($salt.$pass.$salt) [Hashcat Mode: 4710]
[+] Cisco Type 7
[+] BigCrypt
Analyzing 'ECprxlyOyUshcl60I9RBSQ==:3NfdMsGy2AAHhm27PTFe0A=='
[+] Unknown hash
Analyzing '0x102a6bc65c8ec94b21725eb423d44149:dcd7dd32c1b2d80007866dbb3d315ed0'
[+] Unknown hash
Analyzing '$2y$12$RjmAST8RecCMVv1DO4g9zOv3wQ4/vsHlWyC7FLBF/07WL9SzVeA.m'
[+] Blowfish(OpenBSD) [Hashcat Mode: 3200]
[+] Woltlab Burning Board 4.x
[+] bcrypt [Hashcat Mode: 3200]
Analyzing 'GvkWX7r0M7mkLxQ743qaDQ==:0/xjqfLDZfAaeI5/s6zjlg=='
[+] Unknown hash
Analyzing '0xf7c3bc1d808e04732adf679965ccc34ca7ae3441'
[+] BigCrypt
Analyzing '$2y$10$NqnZ8O1xz0UdArMymrPjoOy2.WwLVqfFYxc1rkwwZdw9.f140Wq72'
[+] Blowfish(OpenBSD) [Hashcat Mode: 3200]
[+] Woltlab Burning Board 4.x
[+] bcrypt [Hashcat Mode: 3200]
Analyzing 'qoIP+jhQg+rY3SWAR+FdnZC/H0Gj86WoehIejMC71so='
```

```
[+] Unknown hash
Analyzing '5ts7TXFWp/S2WqMEDXZXO/eEkIE='
[+] PeopleSoft [Hashcat Mode: 133]
Analyzing '64iGbek1bgraDo3VoxjOY4ftjKQj5yWujhG7UyDMCS4='
[+] Unknown hash
Analyzing '$2y$10$fP0pAnFtqOaiOTfjLRGB1.rJU5RE1r/P1xcux8vHd6e1Zw2bJXe7y'
[+] Blowfish(OpenBSD) [Hashcat Mode: 3200]
[+] Woltlab Burning Board 4.x
[+] bcrypt [Hashcat Mode: 3200]
Analyzing 'Administrator:500:aad3b435b51404eeaad3b435b51404ee:957ed2442e3fece7bf7cfe1417115710:::'
[+] Unknown hash
```

As we can see not all hashes could be identified with `hashid`. Additional research is needed to identify the missing hash types.

2. Cracking attempts

Example 1

- Hash: `5baa61e4c9b93f3f0682250b6cf8331b7ee68fd8`
- Password: `password`
- Hash type: SHA1

Password was found via Google on md5calc.com.

Example 2

- Hash: `ECprxlyOyUshcl60I9RBSQ==:3NfdMsGy2AAHhm27PTFe0A==`
- Password:
- Hashtype: `Base64(unhex(md5(password))):Base64(unhex(md5(salt)))`

Based on analysis from https://hashes.com/en/tools/hash_identifier and cracking attempts to identify the order of hash and salt.

Example 3

- Hash: `0x102a6bc65c8ec94b21725eb423d44149:dcd7dd32c1b2d80007866dbb3d315ed0`
- Password:
- Hash type: `md5(password):md5(salt)`
- This is the same as the previous example, but the hash is in hexadecimal.
-

Example 4

- Hash: `$2y$12$RjmAST8RecCMVv1DO4g9zOv3wQ4/vsHlWyC7FLBF/07WL9SzVeA.m`
- Password: `password`
- Hash type: bcrypt

The hash was identified and recovered using hashcat and the “rockyou.txt” wordlist.

Example 5

- Hash: `GvkWX7rOM7mkLxQ743qaDQ==:0/xjqfLDZfAaeI5/s6zjlg==`
- Password:
- Hash type: `Base64(unhex(md5(password))):Base64(unhex(md5(salt)))`

Based on analysis from https://hashes.com/en/tools/hash_identifier and cracking attempts to identify the order of hash and salt.

Example 6

- Hash: `0xf7c3bc1d808e04732adf679965ccc34ca7ae3441`
- Password: `123456789`
- Hash type: SHA1

The hash was found via Google on md5calc.com. Even though Hashid identified the hash as Bigcrypt, the hash appears to be a SHA1 hash.

Example 7

- Hash: `$2y$10$NqnZ8O1xzOUdArMymrPjoOy2.WwLVqfFYxc1rkwwZdw9.f14OWq72`
- Password: `password2020`
- Hash type: bcrypt

The hash was identified by hashid and cracked with hashcat and the top 1000000 passwords wordlist.

Example 8

- Hash: `qoIP+jhQg+rY3SWAR+FdnZC/HOGj86WoehIejMC71so=` (Salt: 1, N 2048, R 8, P 1, Length 32)
- Password: `hugo`
- Hash type: scrypt, saved in Base64

Passing the parameters into Google the type of hash was identified as scrypt. Scrypt hashes can be cracked with hashcat by converting them to the following format: `SCRYPT:n:r:p:salt:hash` so in this case: `SCRYPT:2048:8:1:MQ==:qoIP+jhQg+rY3SWAR+FdnZC/HOGj86WoehIejMC71so=`. Using the "rockyou.txt" the hash was cracked.

Example 9

- Hash: `5ts7TXFWp/S2WqMEDXZXO/eEkIE=`
- Password:
- Hash type: `PepopleSoft`

Hashid identified hash type as PeopleSoft, Dictionary atttack as well as brute-force didn't yield any result.

Example 10

- Hash: `64iGbek1bgraDo3VoxjOY4ftjKQj5yWujhG7UyDMCS4=` (Salt: 2, N 2048, R 8, P 1, Length 32)
- Password:
- Hash type: scrypt, saved in Base64

Passing the parameters into Google the type of hash was identified as scrypt.

Example 11

- Hash: `$2y$10$fP0pAnFtqOaiOTfjLRGB1.rJU5RE1r/P1xcux8vHd6e1Zw2bJXe7y`
- Password:
- Hash type: bcrypt, Blowfish

Hash identified by hashid.

Example 12

- Hash: `Administrator:500:aad3b435b51404eeaad3b435b51404ee:957ed2442e3fece7bf7cfe1417115710:::`
- Password: `CLA80`
- Hash type: Dump from the Windows user database, with a blank LM hash, and a real NT hash

The type of hash was identified by its characteristic structure. Searching for the LM and NT part of the hash the blank LM hash was found on yougottohackthath.com. The NT part of the hash was recovered using ntlm.pw.