CSE877 – Cryptography and Computer Security

University of Nebraska, Lincoln
Fall 2015 – Homework 3
Suraj Ketan Samal
ssamal@cse.unl.edu
(BBUsername: ssamal2)
November 11, 2015

1. [40+(20) points] Implement one of the following Hash functions in the high-level programming language of your choice: MD4, MD5, SHA-1, SHA-256, or SHA-3. Make your implementation callable via the command line by providing a file name (the contents of which will be hashed). The resulting digest should be output to the standard output. Note: you may use online tools or other implementations to debug and test your implementation. You may reference the code of other implementations to help you debug your code as long as the referenced code is in a different language than you are implementing it in.

Bonus Points: You will get 20 bonus points if you implement two Hash functions:one in set {MD4, MD5}, the other one in set {SHA-1, SHA-256, SHA-3}.

Solution:

I have implemented two Hash functions, MD5 (as per RFC 1321) and SHA-1 (as per RFC 3174). Both are located in the source archive submitted as per the following table:

Source File(s)	Location	Description
MyMD5.java	org.unl.cryptoanalysis.tools	MD5 implementation.
MySHA1.java	org.unl.cryptoanalysis.tools	SHA-1 implementation.
Prob1.java	org.unl.cryptoanalysis.hw3	Problem 1(can be used to invoke the md5 and sha-1 implemenations and tests)
build.xml		Ant script for building and testing

The implementations can be run as follows:

user@host> java -jar ./crypto.jar 1 <inputfile> md5
user@host> java -jar ./crypto.jar 1 <inputfile> sha1

Basic Tests can be run using:

user@host> ant test
user@host> java -cp ./crypto.jar org.unl.cryptoanalysis.tools.Tests

2. [20 points] In this exercise, you'll use a library to generate rainbow tables for various hash functions for dictionary-based passwords. Use the standard american dictionary file on CSE (located in the file /usr/share/dict/american which contains 305,089 words). Then, generate rainbow tables for MD4, MD5, SHA-1, SHA-256, and SHA3 (Keccak-256 version), PBKDF2 (using the salt, shoop, r = 1000). Your rainbow table files should have the format: hash:password one per line. Finally, they should be presorted lexicographically using the hash for easy look-up via binary search. You may/should collect the resulting files into one compressed zip file for ease of handing them in.

Solution:

I used a third party library found at (https://github.com/kocakosm/pitaya) that provided all the required implementations (MD4,MD5,SHA-1,SHA-256,SHA3 and PBKDF2). The source files and the generated tables can be found as per the following table :

Source File(s)	**Location**	**Description**
Prob2.java	org.unl.cryptoanalysis.hw3	Problem 2 Implementation.
HashComparator.java	org.unl.cryptoanalysis.tools	Used for sorting while writing records to the rainbow tables file.
Dictionary file on cse.unl.edu (american)	/usr/share/dict/american	Standard dictionary used to generate the rainbow tables.
md4.txt	data/	MD4 rainbow table sorted by hashes.
md5.txt	data/	MD5 rainbow table sorted by hashes.
sha1.txt	data/	SHA1 rainbow table sorted by hashes.
sha256.txt	data/	SHA256 rainbow table sorted by hashes.
sha3.txt	data/	SHA3(Keccak-256) rainbow table sorted by hashes.
pbkdf2.txt	data/	PBKDF2 rainbow table sorted by hashes. (1000 iterations, salt = shoop, output = 20 bytes, Algorithm = HMAC_SHA1)
build.xml		Ant script for building

Since, the output rainbow table files (rtables.zip) were huge, I have archived and put it at two locations below:

a) CSE Server : http://cse.unl.edu/~ssamal/crypto/rtables.zip
b) Google Drive: https://drive.google.com/file/d/0B4A4VgzW2NCqdEJzQ1I1a1lqNzA/view?pli=1

The implementation can be run as follows:

user@host> java -jar ./crypto.jar 2
OR
user@host> java -cp ./crypto.jar org.unl.cryptoanalysis.hw3.Prob2

3. [40 points] We've provided several files containing usernames and hashed passwords using MD4, MD5, SHA1, SHA-256, SHA3 (Keccak 256 version), and PBKDF2 (salt: cse477, 10,000 rounds). You will break as many of these passwords as you can using any means and methods you have at your disposal. You could use your rainbow tables, brute-force strategies, online tools, or dedicated password breaking software such as John the Ripper. Break as many of the passwords as you can and document how each one was broken (what techniques or tools broke each password and how fast).

Solution:

The various resources used to crack the hashes are described below:

Resources	**Description**	**Time Taken**
Existing rainbow tables from (/usr/share/dict/american) and extra compiled (extradict)	Was helpful for all types of hashes.	Few seconds to search once the tables are generated.
HashKiller https://hashkiller.co.uk/md5-decrypter.aspx	An online database of various types of passwords.(was helpful for md5 and sha1).	Instant (within seconds)
CrackStation https://crackstation.net/	An online database of various types of passwords.(was helpful for md4, md5, sha1, sha256).	Instant (within seconds)
John-the-Ripper Community enhanced version http://www.openwall.com/john/ (john-1.8.0-jumbo-1)	Was useful for md4, md5, sha1, sha256, sha3(Keekak) but was not helpful for pbkdf2. **(it was the only external tool helpul for sha3)**	Unable to measure exact time, few md4/md5 passwords were cracked instantly. sha1/sha256/sha3 takes more time.

The code for the implementation is located as under:

Source File(s)	Location	Description
Prob3.java	org.unl.cryptoanalysis.hw3	Problem 3 Implementation.
HashComparator.java	org.unl.cryptoanalysis.tools	Used for sorting while writing records to the rainbow tables file.
SearchTables.java	org.unl.cryptoanalysis.tools	Used to search in the appropriate rainbow table and return back results.
Extra Dictionary	data/extradict	Extra dictionary compiled based on various online tools that helped in cracking the passwords.
Existing hash tables md4.txt md5.txt sha1.txt sha256,txt sha3.txt pbkdf2-cse477.txt	data/	Rainbow tables generated earlier from /usr/share/dict/american.
Extra tables generated md4extra.txt md5extra.txt sha1extra.txt sha256extra,txt sha3extra.txt pbkdf2-cse477extra.txt	data/	Rainbow tables generated from extra dictionary
build.xml		Ant script for building and testing

The implementation looks for the rainbow tables at “data”, if they are not present, it generates them again which may take a while. Hence, it is advised to download the existing rainbow tables archive(rtables.zip) mentioned earlier in (Prob 2) and place it within the data folder before running the program. It also outputs the cracked passwords and a summary.

The implementation can be run as follows:

```
user@host> java -jar ./crypto.jar 3
OR
user@host> java -cp ./crypto.jar org.unl.cryptoanalysis.hw3.Prob3
```

Following results are obtained when the program is executed:

```
####################################
```

```
Problem 3 - Break Hashes
(MD4,MD5,SHA-1,SHA-256 SHA-3(Keekak 256),PBKDF2
####################################

[Using rainbow tables created from ]......
Dictionary: /usr/share/dict/american
Type           Passwords file       Rainbow Table
MD4             data/passwd.md4      data/md4.txt
MD5             data/passwd.md5      data/md5.txt
SHA-1           data/passwd.sha1     data/sha1.txt
SHA-256         data/passwd.sha256   data/sha256.txt
SHA-3           data/passwd.sha3     data/sha3.txt
PBKDF2HMACSHA1  data/passwd.pbkdf2   data/pbkdf2-cse477.txt

==MD4 Results (sorted by username) ==
arizzo:infotaining
bjackson:NOT FOUND
dbarney:NOT FOUND
ejackson:NOT FOUND
jarrieta:NOT FOUND
jgrimm:NOT FOUND
jhammel:NOT FOUND
jlake:NOT FOUND
jrussell:NOT FOUND
jsamardzija:fundamentalist
lvalbuena:scrummy
mszczur:NOT FOUND
nramirez:pulldown
nscheirholtz:terrified
pstrop:NOT FOUND
rkalish:stillings
rrenteria:aardvark
scastro:NOT FOUND
twood:lifeguarding
wcastillo:multiplets
9 out of 20 cracked.(45.0% success)
in 323 msecs

==MD5 Results (sorted by username) ==
bmayhew:antiflu
cortega:NOT FOUND
dchow:NOT FOUND
eschwartz:braininess
```

gfring:pretermination
gschwartz:mylonitic
harchuleta:NOT FOUND
hsalamanca:fundamentalist
hschrader:NOT FOUND
jpinkman:NOT FOUND
mehrmantraut:retail
mschrader:sanctimonies
sgomez:NOT FOUND
sgoodman:NOT FOUND
spete:NOT FOUND
swhite:NOT FOUND
talquist:NOT FOUND
tkitt:scumbering
tsalamanca:NOT FOUND
wwhite:zyzzyvas
9 out of 20 cracked.(45.0% success)
in 173 msecs

==SHA-1 Results (sorted by username) ==
araanta:NOT FOUND
ashaw:nurl
bbickell:NOT FOUND
bsaad:NOT FOUND
ccrawford:NOT FOUND
jmorin:NOT FOUND
jnordstrom:NOT FOUND
joduya:NOT FOUND
jquenneville:NOT FOUND
jtoews:NOT FOUND
kversteeg:neuromuscular
mcarey:raging
mhandzus:undiscriminating
mkruger:NOT FOUND
nhjalmarsson:vulcanizates
nleddy:adscripts
pkane:password
pregin:NOT FOUND
psharp:tweedlers
sbrookbank:NOT FOUND
8 out of 20 cracked.(40.0% success)
in 193 msecs

==SHA-256 Results (sorted by username) ==
agreen:pocketable
arodrigquez:throbber
bchapek:NOT FOUND
bnickens:soberize
bqvale:NOT FOUND
cevans:NOT FOUND
cjzimmerer:NOT FOUND
cpensick:barebones
jankrah:rehearsed
jlong:shakier
jsirles:NOT FOUND
qenunwa:seascape
rkellogg:sneezewood
scriss:NOT FOUND
sjameson:NOT FOUND
sjeanbaptiste:NOT FOUND
slong:NOT FOUND
tmartinez:aberrating
trandle:NOT FOUND
wrichards:NOT FOUND
9 out of 20 cracked.(45.0% success)
in 153 msecs

==SHA-3 Results (sorted by username) ==
acoakley:NOT FOUND
bmack:spacemen
cfraser:NOT FOUND
clundgren:NOT FOUND
dhoward:uvea
fchance:NOT FOUND
hsteinfeldt:NOT FOUND
hzimmerman:sifts
jevers:NOT FOUND
jhayden:waesuck
jpfiester:NOT FOUND
jsheckard:skivered
jslagle:NOT FOUND
jtinkler:NOT FOUND
kdurbin:farmworker
mbrown:unbolt
ooverall:optometers
rkroh:NOT FOUND

```
shofman:NOT FOUND
wschulte:NOT FOUND
8 out of 20 cracked.(40.0% success)
in 176 msecs

==PBKDF2HMACSHA1 Results (sorted by username) ==
bananaman:NOT FOUND
billy:NOT FOUND
bmo:NOT FOUND
cake:NOT FOUND
fionna:NOT FOUND
fmertens:NOT FOUND
iceking:NOT FOUND
jake:NOT FOUND
lemongrab:NOT FOUND
lemonhope:NOT FOUND
lich:NOT FOUND
lsp:NOT FOUND
maja:NOT FOUND
mercelene:NOT FOUND
nester:NOT FOUND
pbubblegum:NOT FOUND
pbutler:NOT FOUND
pward:NOT FOUND
spetrikov:NOT FOUND
treetrunks:NOT FOUND
0 out of 20 cracked.(0.0% success)
in 2 msecs
=========RainbowTable Results==========
Type          Total     Cracked   Sucess(%)  Time(ms)
MD4           20        9         45.0       323
MD5           20        9         45.0       173
SHA-1         20        8         40.0       193
SHA-256       20        9         45.0       153
SHA-3         20        8         40.0       176
PBKDF2HMACSHA1  20        0         0.0        2

[Using rainbow tables created from ].....
Dictionary: data/extradict
Type          Passwords file      Rainbow Table
MD4           data/passwd.md4     data/md4extra.txt
MD5           data/passwd.md5     data/md5extra.txt
SHA-1         data/passwd.sha1    data/sha1extra.txt
```

```
SHA-256        data/passwd.sha256   data/sha256extra.txt
SHA-3          data/passwd.sha3     data/sha3extra.txt
PBKDF2HMACSHA1  data/passwd.pbkdf2   data/pbkdf2-cse477extra.txt

==MD4 Results (sorted by username) ==
arizzo:infotaining
bjackson:8afs
dbarney:NOT FOUND
ejackson:NOT FOUND
jarrieta:obama
jgrimm:NOT FOUND
jhammel:NOT FOUND
jlake:NOT FOUND
jrussell:NOT FOUND
jsamardzija:fundamentalist
lvalbuena:scrummy
mszczur:NOT FOUND
nramirez:pulldown
nscheirholtz:terrified
pstrop:tbj$
rkalish:stillings
rrenteria:aardvark
scastro:987654321
twood:lifeguarding
wcastillo:multiplets
13 out of 20 cracked.(65.0% success)
in 1 msecs

==MD5 Results (sorted by username) ==
bmayhew:antiflu
cortega:NOT FOUND
dchow:696d29e0940a4957748fe3fc9efd22a3
eschwartz:braininess
gfring:pretermination
gschwartz:mylonitic
harchuleta:michaeljordan
hsalamanca:fundamentalist
jpinkman:Kx69
mehrmantraut:retail
mschrader:sanctimonies
sgomez:NOT FOUND
sgoodman:sSDD
spete:X#334
```

swhite:NOT FOUND
talquist:Tr0ub4dor&3
tkitt:scumbering
tsalamanca:/,*SYE|
wwhite:zyzzyvas
16 out of 20 cracked.(80.0% success)
in 2 msecs

==SHA-1 Results (sorted by username) ==
araanta:4pXR
ashaw:nurl
bbickell:PasswordIsTaco
bsaad:NOT FOUND
ccrawford:NOT FOUND
jmorin:e?mdls^
jnordstrom:NOT FOUND
joduya:NOT FOUND
jquenneville:NOT FOUND
jtoews:NOT FOUND
kversteeg:neuromuscular
mcarey:raging
mhandzus:undiscriminating
mkruger:#N4/
nhjalmarsson:vulcanizates
nleddy:adscripts
pkane:password
pregin:pleaseletmein
psharp:tweedlers
sbrookbank:zoPJd6f
14 out of 20 cracked.(70.0% success)
in 2 msecs

==SHA-256 Results (sorted by username) ==
agreen:pocketable
arodrigquez:throbber
bchapek:NOT FOUND
bnickens:soberize
bqvale:NOT FOUND
cevans:NOT FOUND
cjzimmerer:NOT FOUND
cpensick:barebones
jankrah:rehearsed
jlong:shakier

jsirles:J-M?
qenunwa:seascape
rkellogg:sneezewood
scriss:NOT FOUND
sjameson:00000000
sjeanbaptiste:NOT FOUND
slong:1234567890
tmartinez:aberrating
trandle:NOT FOUND
wrichards:GS=$
13 out of 20 cracked.(65.0% success)
in 2 msecs

==SHA-3 Results (sorted by username) ==
acoakley:NOT FOUND
bmack:spacemen
cfraser:NOT FOUND
clundgren:NOT FOUND
dhoward:uvea
fchance:NOT FOUND
hsteinfeldt:NOT FOUND
hzimmerman:sifts
jevers:CjIZd
jhayden:waesuck
jpfiester:NOT FOUND
jsheckard:skivered
jslagle:12345
jtinkler:SvFP
kdurbin:NOT FOUND
mbrown:unbolt
ooverall:NOT FOUND
rkroh:NOT FOUND
shofman:NOT FOUND
wschulte:NOT FOUND
9 out of 20 cracked.(45.0% success)
in 2 msecs

==PBKDF2HMACSHA1 Results (sorted by username) ==
bananaman:PasswordIsTaco
billy:NOT FOUND
bmo:throbber
cake:NOT FOUND
fionna:Tr0ub4dor&3

```
fmertens:barebones
iceking:NOT FOUND
jake:infotaining
lemongrab:NOT FOUND
lemonhope:password
lich:NOT FOUND
lsp:NOT FOUND
maja:8afs
mercelene:NOT FOUND
nester:NOT FOUND
pbubblegum:pretermination
pbutler:NOT FOUND
spetrikov:NOT FOUND
treetrunks:NOT FOUND
8 out of 20 cracked.(40.0% success)
in 2 msecs
=========Final Results==========
Type            Total      Cracked    Success(%)
MD4             20         13         65.0
MD5             20         16         80.0
SHA-1           20         14         70.0
SHA-256         20         13         65.0
SHA-3           20         9          45.0
PBKDF2HMACSHA1  20         8          40.0
```
