Assignment-5- Optimization of PBKDF

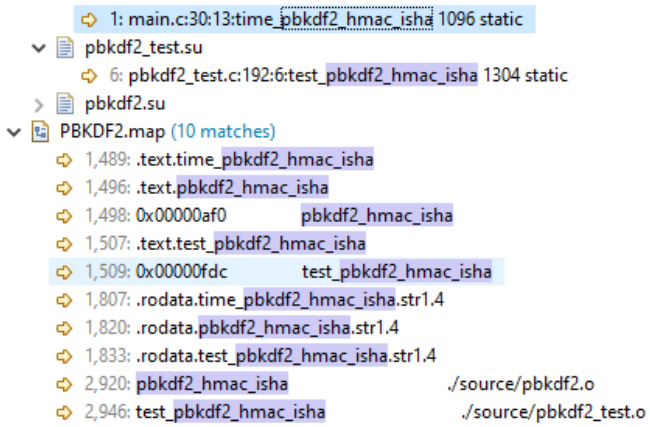
Author: Swapnil Ghonge

Techinical Memo of the Optimization

pbkdf2_hmac_isha() Function:-

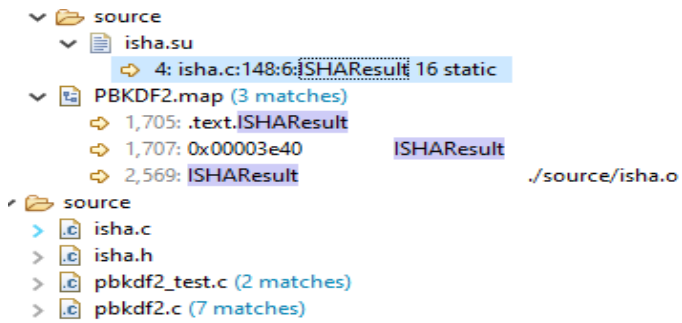
ISHAReset Function()

This function resets all the parameters to 0 such as length_low , Length_high. Additionally It also assigns hex values to ctx->MD[0] to ctx->MD[4]. Moreover it also initializes computed and corrupted values.



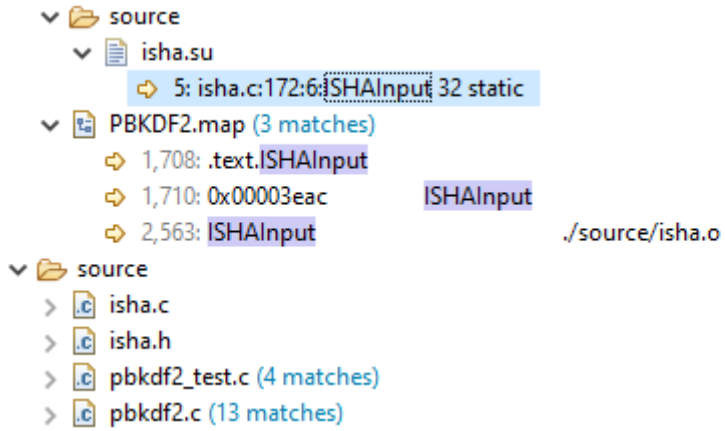
ISHAResult Function:-

This function displays the output of all the bytes goes through the algorithm. The data of digest (MD[i]) are stored as 20 bytes in digest out after performing bitswap operation. Additionally ISHAPadMessage is called and the computed flag is set.



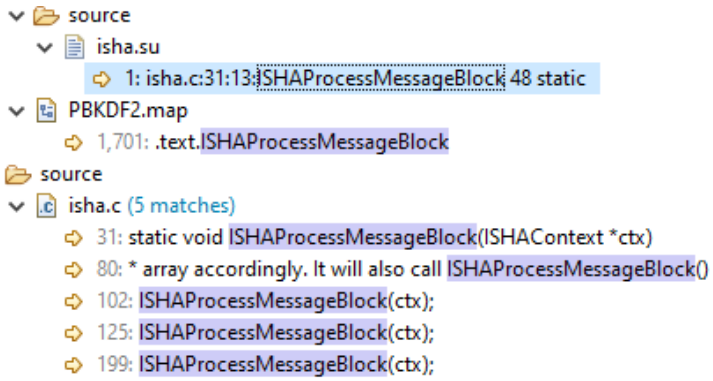
ISHAInput Function:-

This function is used to take input for the hashing function. It takes 3 parameters as inputs such as context, message array and length. Firstly it checks for the length errors. Then the byte is stored in the data structure of context in the variable buffer_len then it is incremented everytime by the value length. The loops runs till the difference of length and temp is zero, the value message array is copied in ctx->MBlock. After the loop is execuetedthe ISHAProcessMessageBlock is called when index of context is equal to 64 bytes of ISHA.

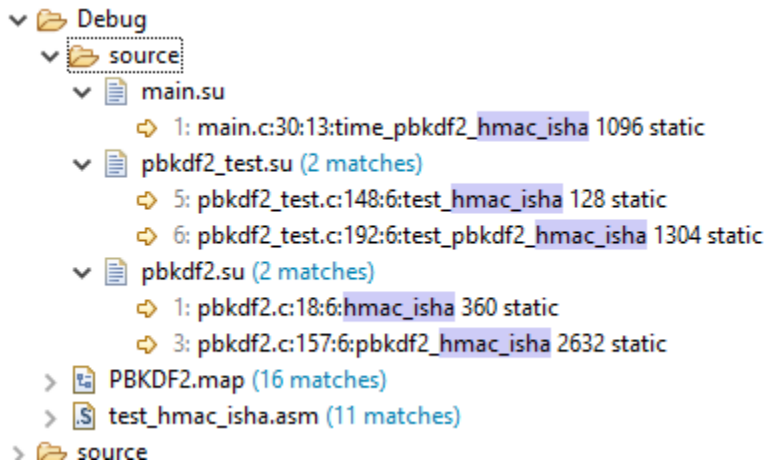


ISHAPadMessage Function:-

The message must be padded to an even 512 bits. The first padding bit must be a '1'. The last 64 bits represent the length of the original message. All bits in between should be 0. This function will pad the message according to those rules by filling the MBlock array accordingly. It will also call ISHAProcessMessageBlock() appropriately. When it returns, it can be assumed that the message digest has been computed.



hmac_isha Function()
This function takes 5 inputs key pointer,key_len, msg pointer, msg_len and digest.
It Computes the HMAC-ISHA for the given key and message. *
Parameters:
key: The secret key
key_len: Length of key
msg: The message to be hashed
msg_len: Length of msg
digest: Output area: the 20-byte digest will be written here
Returns:
20-byte computed key is returned in digest



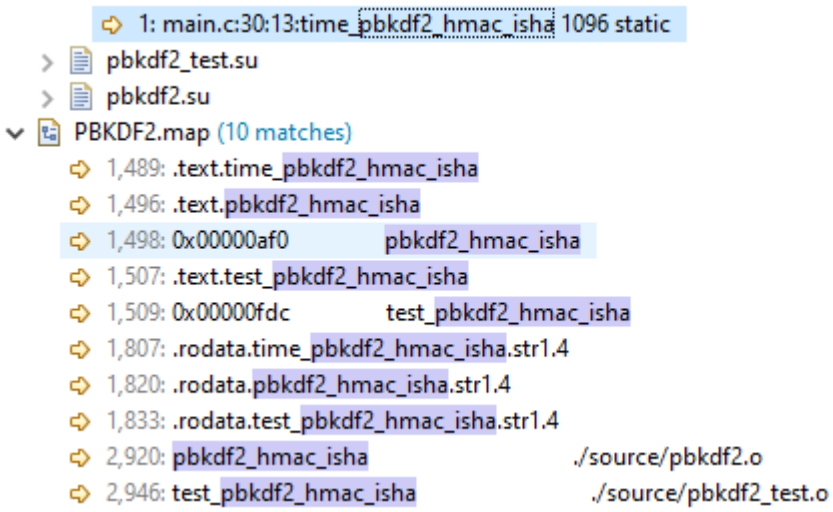
pbkdf2_hmac_isha function:-

Parameters:

* pass The password

* pass_len length of password
* salt The salt
* salt_len length of salt
* iter The iteration count ("c" in RFC 8018)
* dkLen intended length in bytes of the derived key
* DK the output derived key, must be at least dkLen bytes long

An ‘I’ is calculated to find the number of 20 is to 8 bits blocks. The Function ‘F’ is called for each block of DK where pass, pass_len, salt, salt_len, iteration count and block index are passed to compute the block. It combines the blocks and gives the first dkLen octets to make a derived key DK. The resultant data is stored in DK. The derived key of ‘dkLen’ bytes is stored in DK.

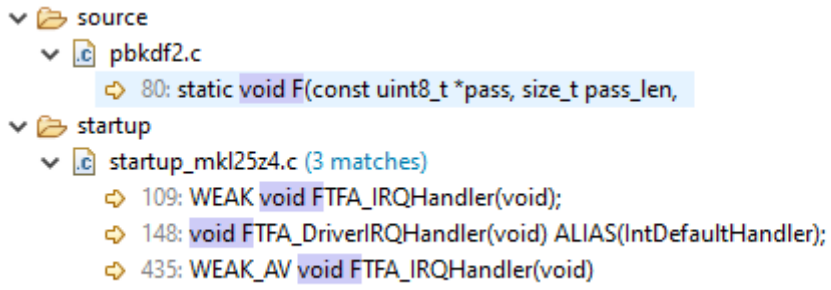


F Function:-

This function takes 7 input namely password, password length, salt, salt length, iterations, result and the block index.

The data of ‘salt’ is copied into ‘saltplus’ and block index is appended in 4 bytes big endian by using bswap operations as ARM is little endian. Function hmack_isha() has been called inside this F() function with pass, pass_len, saltplus, salt_len passed as parameters. This function array hashes the ‘salt’ and ‘pass’ to

make it into block length array. This iteration is added at the end of the array and hashing is done 4096 times



Timing Analysis:

Approximate time take by function:

Function	Time taken msec
F()	2914
pbkdf2_hmac_isha()	8744
hmac_isha()	0.711
ISHAReset()	0.024
ISHAResult()	0.013
ISHAInput()	0.1626
ISHAProcessMessageBlock()	0.058
SHAPadMessage()	0.086

Number of Calls to the function

Number of calls	Calls
pbkdf2_hmac_isha()	1
F()	3
hmac_isha()	12288
ISHAReset()	24576
ISHAResult()	24576
ISHAInput()	49152
ISHAProcessMessageBlock()	49152
ISHAPadMessage()	24576

Size of text segment before Optimization:

```
Performing post-build steps
arm-none-eabi-size "PBKDF6.axf"; # arm-none-eabi-objcopy -v -O binary "PBKDF6.axf"
   text    data     bss     dec     hex filename
  21056       8    9724   30788    7844 PBKDF6.axf
```

Size of text segment after Optimization:

```
make --no-print-directory post-build
Performing post-build steps
arm-none-eabi-size "PBKDF2.axf"; # arm-none-eabi-objcopy -v -O binary "PBKDF2.axf"
   text    data     bss     dec     hex filename
  21056       8    9724   30788    7844 PBKDF2.axf
```

Timing before Optimization: 8744msec

```
Running validity tests...
test_isha test 0: success
test_isha test 1: success
test_isha test 2: success
test_isha test 3: success
test_isha test 4: success
test_isha test 5: success
test_isha test 6: success
test_isha test 7: success
test_hmac_isha test 0: success
test_hmac_isha test 1: success
test_hmac_isha test 2: success
test_pbkdf2_hmac_isha test 0: success
test_pbkdf2_hmac_isha test 1: success
test_pbkdf2_hmac_isha test 2: success
test_pbkdf2_hmac_isha test 3: success
test_pbkdf2_hmac_isha test 4: success
test_pbkdf2_hmac_isha test 5: success
test_pbkdf2_hmac_isha test 6: success
test_pbkdf2_hmac_isha test 7: success
test_pbkdf2_hmac_isha test 8: success
test_pbkdf2_hmac_isha test 9: success
test_pbkdf2_hmac_isha test 10: success
All tests passed!
Running timing test...
time_pbkdf2_hmac_isha: 4096 iterations took 8744 msec
```

Timing after optimization: 2660msec