

Hash Message Authentication Code (HMAC)

Cryptography

Nama: Ramadhan Kalih Sewu

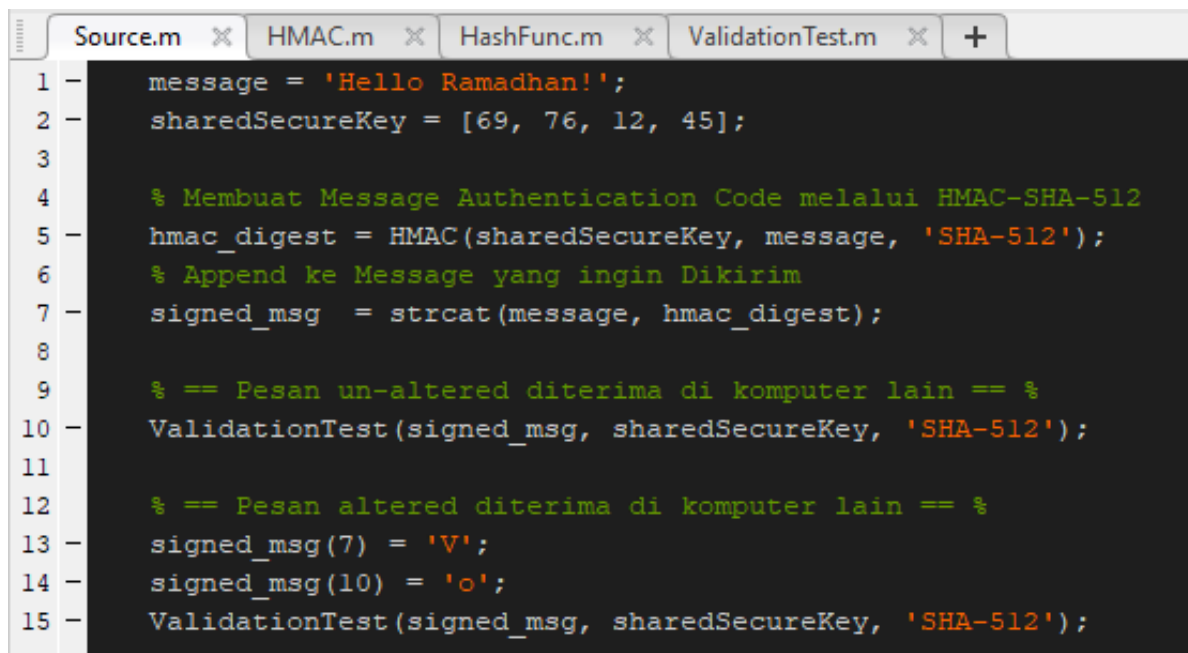
NPM: 1806148826

I. Kunci Simetris

Dalam pengujian algoritma HMAC, kita akan mengirim sebuah pesan dengan dua skenario yaitu pesan tidak diubah dan pesan diubah. Algoritma hash untuk HMAC yang kita gunakan untuk komunikasi harus sama. Dalam kasus ini, kita gunakan SHA-512. Algoritma ini juga memiliki ekspektasi bahwa kunci harus simetris. Oleh karena itu, kita anggap kunci diperoleh oleh kedua belah pihak (pengirim dan penerima) dalam suatu saluran yang aman.

Pesan atau data asli akan ditambahkan dengan kode HMAC sebagai satu 'bundle' dan dikirim melalui saluran yang **tidak** aman. Nilai HMAC akan melindungi kita dari:

- Seseorang mengubah data saat proses transmisi.
- Seseorang menyamar menjadi lawan bicara kita.



```
Source.m x HMAC.m x HashFunc.m x ValidationTest.m x +
1 - message = 'Hello Ramadhan!';
2 - sharedSecureKey = [69, 76, 12, 45];
3
4 % Membuat Message Authentication Code melalui HMAC-SHA-512
5 - hmac_digest = HMAC(sharedSecureKey, message, 'SHA-512');
6 % Append ke Message yang ingin Dikirim
7 - signed_msg = strcat(message, hmac_digest);
8
9 % == Pesan un-altered diterima di komputer lain == %
10 - ValidationTest(signed_msg, sharedSecureKey, 'SHA-512');
11
12 % == Pesan altered diterima di komputer lain == %
13 - signed_msg(7) = 'V';
14 - signed_msg(10) = 'o';
15 - ValidationTest(signed_msg, sharedSecureKey, 'SHA-512');
```

HMAC akan melakukan hash terhadap variabel pesan dan kunci. Hash terhadap pesan akan menjaga integritas dari pesan sehingga kita tahu pesan terjaga keasliannya. Sedangkan, hash terhadap kunci akan menjaga autentikasi sehingga kita dapat mengetahui apakah pesan dikirim dari seseorang yang kita duga.

```
Source.m x HMAC.m x HashFunc.m x ValidationTest.m x +
1 function [hmac] = HMAC(Key, Msg, Method)
2     if strcmp(Method, 'MD5') || strcmp(Method, 'SHA-1') || strcmp(Method, 'SHA-256')
3         BlockSize = 64;
4     else
5         BlockSize = 128;
6     end
7     KeySize = numel(Key);
8     Key = uint8(Key); % Hash it if it is longer than BlockSize
9     ipad(1:BlockSize) = uint8(54); % 0x36
10    ipad(1:KeySize) = bitxor(uint8(54), Key);
11    opad(1:BlockSize) = uint8(92); % 0x5c
12    opad(1:KeySize) = bitxor(uint8(92), Key);
13    Engine2 = java.security.MessageDigest.getInstance(Method);
14    Engine2.update(ipad);
15    Engine2.update(uint8(Msg));
16    iHash = typecast(Engine2.digest, 'uint8');
17    Engine2.update(opad);
18    Engine2.update(iHash);
19    hmac = typecast(Engine2.digest, 'uint8');
20 end
```

Fungsi hash telah tersedia dalam library java dengan beberapa metode seperti SHA-1, SHA-256, SHA-512, MD5, dll. Ini dapat kita gunakan untuk menunjang fungsi HMAC yang memerlukan fungsi hash eksternal.

```
Source.m x HMAC.m x HashFunc.m x ValidationTest.m x +
1 function [md] = HashFunc(Msg, Method)
2     Engine = java.security.MessageDigest.getInstance(Method);
3     Engine.update(uint8(Msg(:)));
4     md = typecast(Engine.digest, 'uint8');
5 end
```

Fungsi ini akan mengecek integritas dan autentikasi data melalui HMAC. Kita akan bandingkan nilai HMAC yang dilampirkan dalam suatu pesan dengan nilai HMAC yang kita kalkulasi ulang menggunakan kunci simetris.

```
1 function [] = ValidationTest(signed_msg, key, method)
2
3     get_message = signed_msg;
4     % Sesuai kesepakatan, kita menggunakan algoritma yang sama HMAC-SHA-512
5     % oleh karena itu, kita tahu panjang padded message sebanyak 64 byte
6     hmac = get_message(end-63:end);
7     only_message = get_message(1:end-64);
8     % kalkulasi nilai HMAC
9     hmac_test = HMAC(key, only_message, method);
10
11     fprintf('%s -> ', only_message);
12     if (hmac == hmac_test) display('Valid');
13     else display('Invalid');
14     end
15
16 end
```

Output menunjukkan bahwa HMAC dapat mendeteksi bahwa suatu pesan telah diubah. Dalam aplikasinya, kunci simetris harus dikirim melalui saluran yang aman. Jika hacker mengetahui algoritma hash yang digunakan dan berhasil mencuri kunci simetris, ia dapat melakukan perubahan pada data dan kemudian menghitung ulang nilai HMAC-nya. Dengan ini, hacker dapat dengan sukses menyamar sebagai pihak berwenang. Kelemahan ini bertumpu pada kunci simetris. Oleh karena itu, kita dapat gunakan metode AES untuk membentuk kunci tersebut dengan aman.'

```
>> Source
Hello Ramadhan! -> Valid
Hello Vamodhan! -> Invalid
```

II. Kunci Asimetris (RSA)

Kita tidak dapat memvalidasi kebenaran HMAC jika tidak menggunakan kunci yang sama, maka kunci simetris wajib hukumnya. Tetapi, HMAC belum memenuhi faktor keamanan non-repudiation. Oleh karena itu, kunci asimetris dapat di gunakan untuk men-enkripsi sebuah pesan agar hanya dapat dilihat oleh si penerima. Dalam kasus ini, Alice adalah penerima dan Bob adalah pengirim.

- Alice membangun RSA
- Alice mengirim Public Key melalui saluran tak aman
- Bob menghitung nilai HMAC dengan kunci shared rahasia.
- Bob menggabungkan pesan dengan nilai HMAC kemudian enkripsi menggunakan public key.
- Alice menerima pesan, kemudian dekripsi dengan private key.
- Alice cek integritas dan keaslian data melalui validasi HMAC dengan kunci shared rahasia.

Pesan yang dikirim melalui saluran tak aman akan dijaga oleh enkripsi. Berikut adalah pesan yang dienkripsi menggunakan RSA:

```
signed_msg =  
  
Columns 1 through 7  
16748651    62366049    79100229    79100229    59437148    34436749    76837821  
  
Columns 8 through 14  
53217255    21560608    53217255    8117808     42126908    53217255    29422925  
  
Columns 15 through 21  
51601636    79100229    2040962     49063261    67996055    3426393     76942766  
  
Columns 22 through 28  
36665747    80510960    63644896    29063501    19422239    81249353    3239441  
  
Columns 29 through 35  
81870088    69989774    28606757    14032246    42781941    29438093    9832227  
  
Columns 36 through 42  
40467687    81399518    49083416    12571689    49942643    36943526    43882513
```

```

1  % [1] == Alice membangun RSA == %
2  p = 8831;
3  q = 9769;
4  % Generate Keys
5  [pub, priv, modulo] = GenerateRSA(p, q);
6  % mengirim kunci public
7  sharedSecureKey = [69, 76, 12, 45];
8
9  % [2] == Bob mengirim pesan == %
10 message = 'Hello Ramadhan!';
11 % Membuat Message Authentication Code melalui HMAC-SHA-512
12 hmac_digest = HMAC(sharedSecureKey, message, 'SHA-512');
13 % Append ke Message yang ingin Dikirim
14 signed_msg = strcat(message, hmac_digest);
15 % Enkripsi dengan Public Key
16 signed_msg = double(signed_msg);
17 for i=1:length(signed_msg)
18     signed_msg(i) = FastExponent(signed_msg(i), pub, modulo);
19 end
20
21 % [3] == Alice menerima pesan un-altered== %
22 % Dekripsi dengan Private Key
23 for i=1:length(signed_msg)
24     signed_msg(i) = FastExponent(signed_msg(i), priv, modulo);
25 end
26 signed_msg = char(signed_msg);
27 % Tes Integritas dan Keaslian data menggunakan HMAC
28 ValidationTest(signed_msg, sharedSecureKey, 'SHA-512');

```

```

>> Source
Hello Ramadhan! -> Valid

```

Kasus pesan diubah:

```

% [EX] == Alice menerima pesan altered == %
signed_msg(7) = 'V';
signed_msg(10) = 'o';
% Dekripsi dengan Private Key
for i=1:length(signed_msg)
    signed_msg(i) = FastExponent(signed_msg(i), priv, modulo);
end
signed_msg = char(signed_msg);
% Tes Integritas dan Keaslian data menggunakan HMAC
ValidationTest(signed_msg, sharedKey, 'SHA-512');

```

```

XXXXXXXXXXXXXXXXXXXX -> Invalid

```