**II4021 Kriptografi**

# Tanda-tangan Digital


Oleh: Rinaldi M

Program Studi Sistem dan Teknologi Informasi
Sekolah Teknik Elektro dan Informatika
ITB - 2026

# Review materi awal

- Ingat kembali empat layanan keamanan yang disediakan oleh kriptografi:
  1. Kerahasiaan pesan (*confidentiality/secrecy*)
  2. Keaslian pesan (*data integrity*).
  3. Otentikasi (*authentication*)
  4. Anti-penyangkalan (*nonrepudiation*).

- Layanan 1 dilakukan dengan mengenkripsi/dekripsi pesan
- Layanan 2 dilakukan dengan fungsi hash dan MAC
- Layanan 3 dilakukan dengan menggunakan MAC dan tanda-tangan digital (*digital signature*).
- Layanan 4 dilakukan dengan menggunakan tanda-tangan digital (*digital signature*).

# Tanda-tangan

- Sejak zaman dahulu, tanda-tangan sudah digunakan untuk otentikasi dokumen cetak.

- Tanda-tangan mempunyai karakteristik sebagai berikut:
  1. Tanda-tangan adalah bukti yang otentik.
  2. Tanda tangan tidak dapat dilupakan.
  3. Tanda-tangan tidak dapat dipindah untuk digunakan ulang.
  4. Dokumen yang telah ditandatangani tidak dapat diubah.
  5. Tanda-tangan tidak dapat disangkal.

Hari: Senin, Tanggal 12 September 2016

Waktu: 14.00 Wib

Tempat: Aula

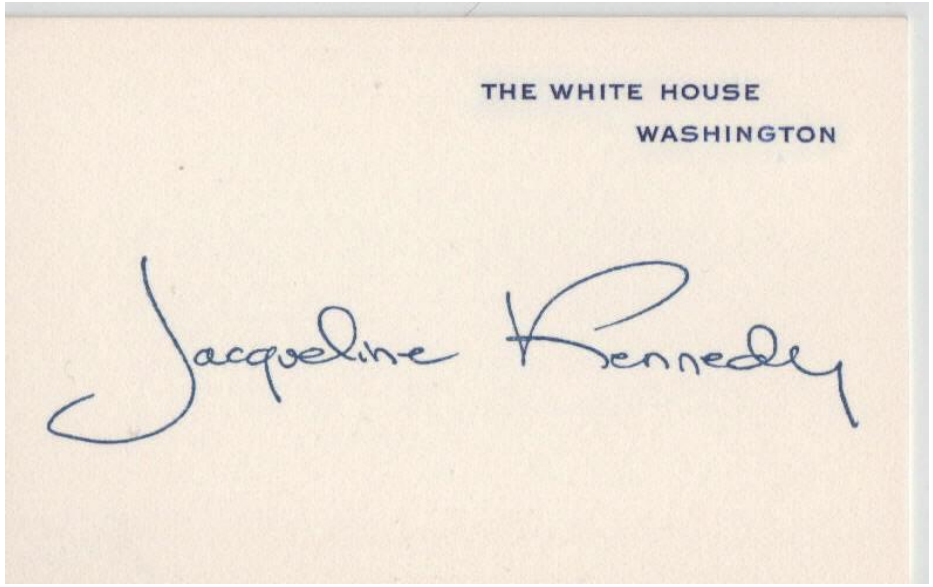
Demikian kami buat surat ini dengan sebenarnya. Harap semua peserta untuk hadir sesuai jadwal yang telah disebutkan di atas. Terima kasih.

Tertanda tangan

Mr Mukidi

Master of Ceremony
Doktor Mukidi, Mpd

- Fungsi tanda tangan pada dokumen kertas juga diterapkan untuk otentikasi pada data digital (pesan, dokumen elektronik).
- Tanda-tangan untuk data digital dinamakan **tanda-tangan digital** (*digital signature*).
- Tanda-tangan digital di dalam konteks kriptografi tidak sama dengan tanda-tangan yang di-digitisasi (*digitized signature*) dengan cara dipindai atau difoto.




*digitized signature*

*digitized signature*

- Tanda-tangan digital adalah *nilai kriptografis* yang bergantung pada isi pesan dan kunci.

- Jika tanda-tangan seseorang pada dokumen cetak selalu sama, apa pun isi dokumennya,

- maka tanda-tangan digital selalu berbeda-beda antara satu pesan dengan pesan lain, dan/atau antara satu kunci dengan kunci yang lain.

Paris, 31 Desember 2018

Halo Alice
Sudah lama kita tidak berjumpa sejak lulus SMA. Saya sekarang tinggal di Paris sejak tahun 2016. Saya bekerja di sebuah perusahaan IT yang bernama Solution Express. Perusahaan ini memberikan layanan keamanan informasi berbasis cloud computing. Saya menjadi menejer Quality Control. Klien kami umumnya adalah bank-bank yang membutuhkan keamanan data nasabah.

Oh ya, saya belum menanyakan bagaimana keadaanmu sekarang. Di mana kamu bekerja atau malah melanjutkan studi S2 di mana? Saya ingat kamu dulu  jago sekali pelajaran kimia. Apakah kamu masih menekuni bidang kimia saat ini?

Oke deh, jika kamu jalan-jalan ke Eropa jangan lupa mampir ke kota Paris. Nanti saya akan ajak kamu mengunjungi Menara Eiffel. Bisa naik sampai ke  atas lho.

Salam dari temanmu di Paris
Bob

-- BEGIN SIGNATURE—
**13706B6D42442620B2FD1098BD4D54ADFA9F7DC27576954ADCE5E5FC901**
-- END SIGNATURE--

# Persyaratan Tanda Tangan Digital

1. Tanda tangan harus berupa rangkaian bit yang bergantung pada pesan yang ditandatangani
2. Tanda tangan  harus menggunakan informasi yang unik (=kunci) dari pengirim untuk mencegah pemalsuan dan penyangkalan
3. Membangkitkan tanda tangan digital harus relatif mudah dilakukan
4. Mengenali dan memverifikasi tanda tangan digital harus relatif mudah dilakukan
5. Secara komputasi hampir tidak mungkin memalsukan tanda tangan digital, baik dengan merekonstruksi pesan baru untuk tanda tangan digital yang sudah ada, atau merekonstruksi tanda tangan curang untuk pesan yang diberikan
6. Menyimpan salinan tanda tangan digital ke dalam storage harus mudah dilakukan secara praktek.

# Dua proses dalam tanda-tangan digital

1. Menandatangani pesan (*signing*)
   Memberi tanda-tangan pada pesan

2. Memverifikasi pesan (*verification*)
   Memeriksa keabsahan tanda-tangan digital

# Bagaimana cara menandatangani pesan?

Ada dua cara yang dilakukan untuk menandatangani pesan:

1. Mengenkripsi pesan

   - Khusus untuk pesan rahasia

2. Menggunakan kombinasi fungsi *hash* (*hash function*) dan kriptografi kunci-publik

   - Untuk pesan yang **tidak perlu** rahasia

# Penandatanganan dengan Cara Mengenkripsi Pesan

## a. Enkripsi menggunakan algoritma kriptografi kunci-simetri

- Pesan yang dienkripsi dengan algoritma simetri sudah memberikan solusi untuk otentikasi pengirim karena kunci simetri hanya diketahui oleh pengirim dan penerima.

- Namun cara ini tidak menyediakan cara untuk melakukan nir-penyangkalan (*non-repudiation*).

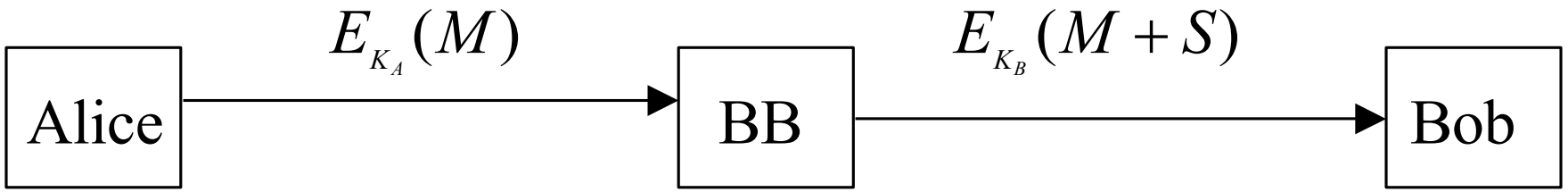
- Jika Alice menyangkal telah mengirim pesan kepada Bob, maka Bob tidak punya cara untuk membantah sangkalan Alice. Alice dapat saja menuduh bahwa pesan tersebut dibuat oleh Bob sendiri karena hanya Alice dan Bob yang mengetahui kunci untuk enkripsi dan dekripsi.

- **Kesimpulan**: menandatangani pesan dengan kriptografi kunci simetri tidak dapat dilakukan.

- Agar kriptografi kunci-simeteri dapat mengatasi masalah penyangkalan, maka diperlukan pihak ketiga yang dipercaya oleh pengirim/penerima.
- Pihak ketiga ini disebut penengah (*arbitrase*).

- Misalkan BB (*Big Brothers*) adalah  otoritas arbitrase yang dipercaya oleh Alice dan Bob.
- BB memberikan kunci rahasia $K_A$ kepada Alice dan kunci rahasia $K_B$ kepada Bob.
- Hanya Alice dan BB yang mengetahui $K_A$, begitu juga hanya Bob dan BB yang mengetahui $K_B$.

- Jika Alice bekirim pesan $M$ kepada Bob, maka langkah-langkahnya adalah sebagai berikut:
  1. Alice mengenkripsi pesan $M$ untuk Bob dengan $K_A$, lalu mengirim cipherteksnya ke BB.
  2. BB melihat bahwa pesan dari Alice, lalu mendekripsi pesan dari Alice dengan $K_A$.
  3. BB membuat pernyataan $S$ bahwa ia menerima pesan dari Alice, lalu menambahkan pernyataan tersebut pada plainteks dari Alice.
  4. BB mengenkripsi bundel pesan $(M + S)$ dengan $K_B$, lalu mengirimkannya kepada Bob.
  5. Bob mendekripsi bundel pesan dengan $K_B$. Ia dapat membaca pesan dari Alice ($M$) dan pernyataan ($S$) dari BB bahwa Alice yang mengirim pesan tersebut.



- Jika Alice menyangkal telah mengirim pesan tersebut, maka pernyataan dari BB pada pesan yang diterima oleh Bob digunakan untuk menolak sangkalan Alice.

- Bagaimana BB tahu bahwa pesan tersebut dari Alice dan bukan dari Charlie?

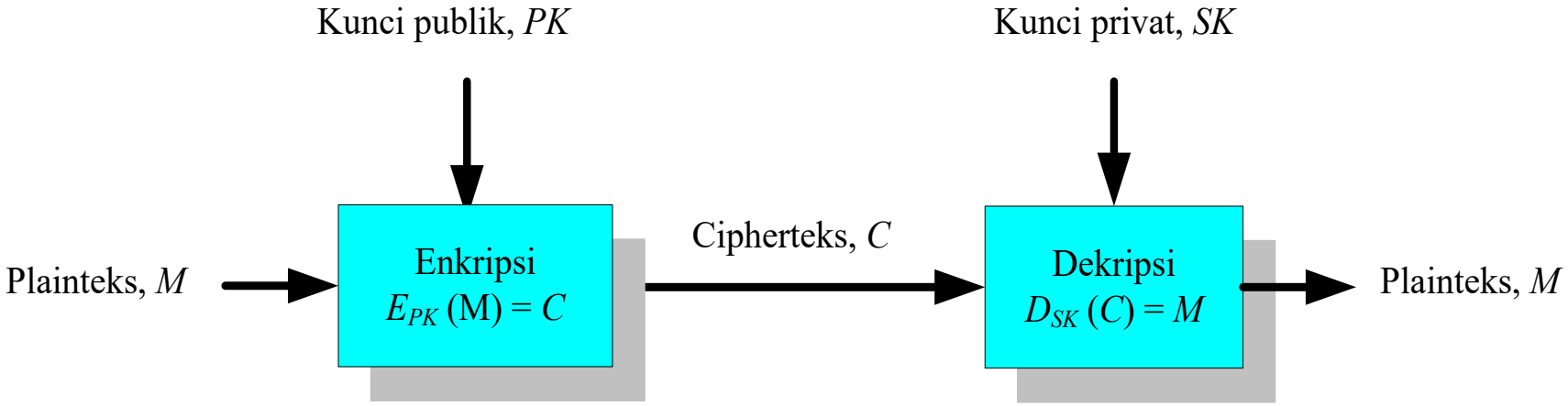
- Karena hanya BB dan Alice yang mengetahui kunci rahasia, maka hanya Alice yang dapat mengenkripsi pesan dengan kunci terebut.

- **Kelemahan**: pelibatan pihak ketiga dalam penandatanganan pesan membuatnya menjadi lebih rumit, tidak praktis, dan tidak sangkil (*efficient*) sehingga tidak lazim digunakan di dalam praktek dunia nyata.

- Solusinya adalah menandatangani pesan dengan menggunakan kriptografi kunci-publik seperti penjelasan berikut ini.

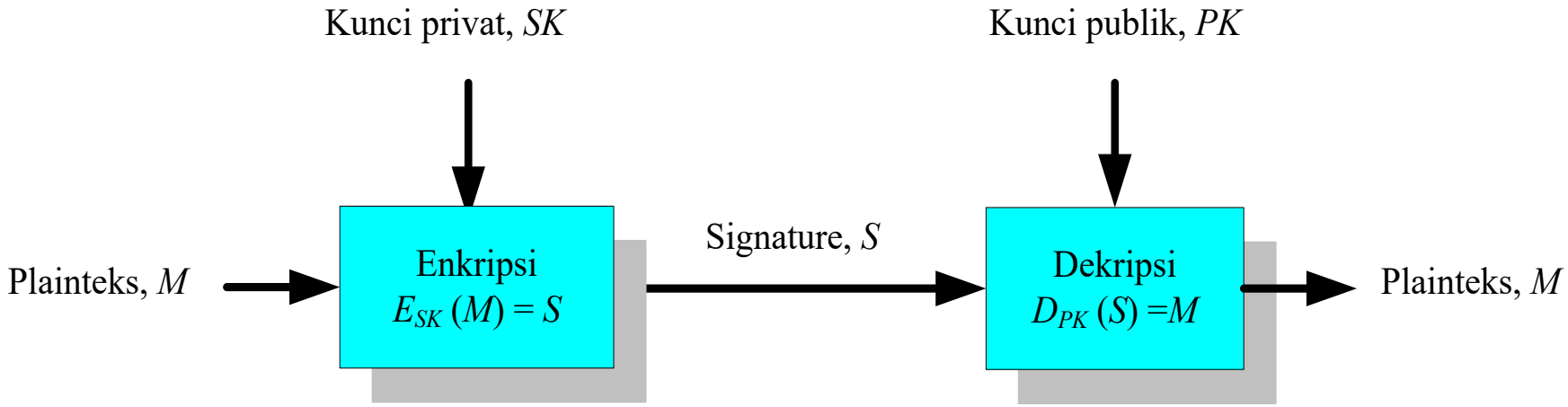
## b. Enkripsi mengunakan algoritma kriptografi kunci-publik

- Mengenkripsi pesan dengan menggunakan kunci publik penerima pesan dan mendekripsinya dengan kunci privat penerima pesan adalah proses yang normal di dalam kriptografi kunci-publik.



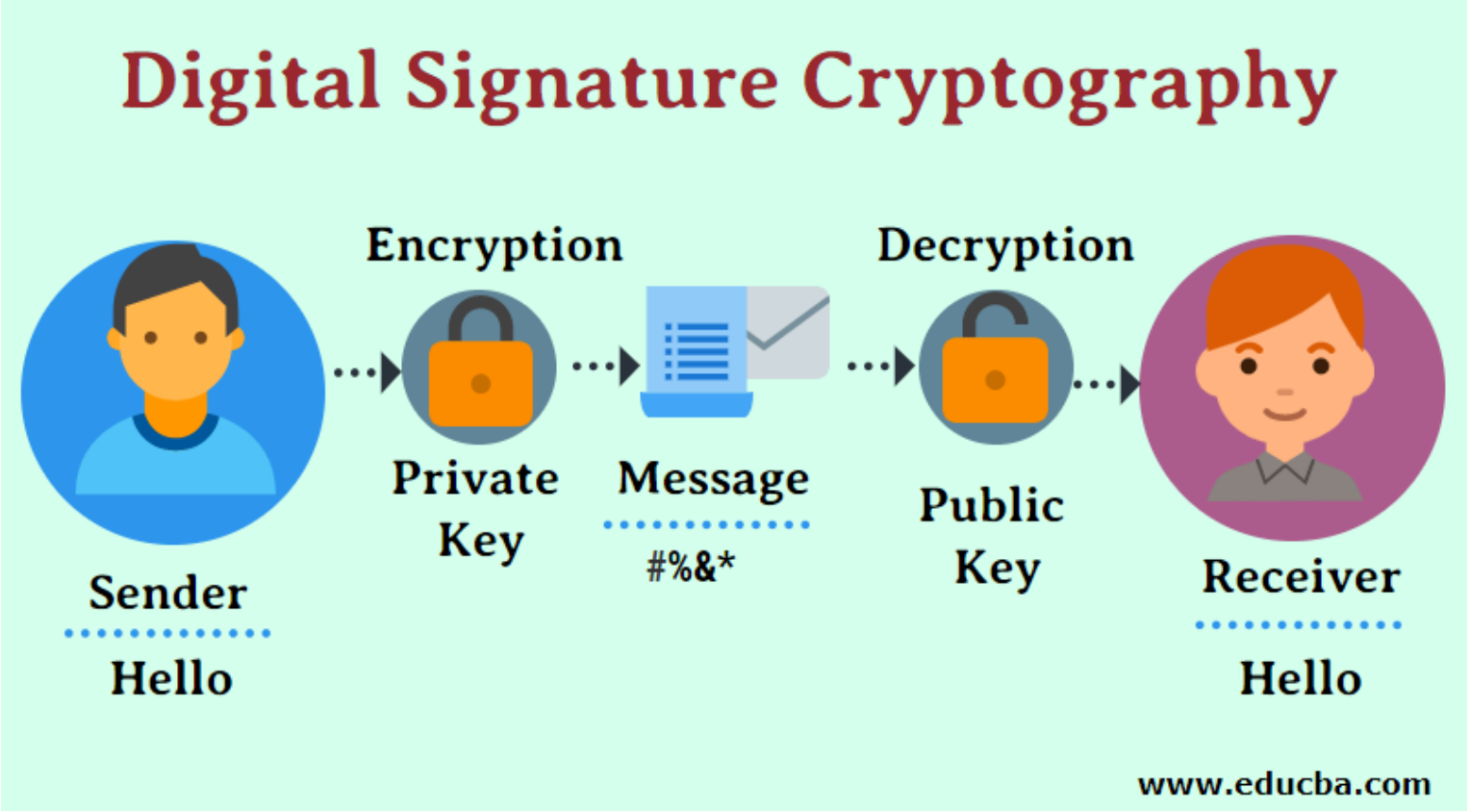

- Namun cara ini tidak dapat mengotentikasi si pengirim pesan karena kunci publik diketahui oleh siapapun.

- Oleh karena itu, agar kriptografi kunci-publik dapat berfungsi sebagai tanda-tangan digital, maka prosesnya dibalik:
  - pesan dienkripsi dengan kunci privat si pengirim.
  - pesan didekripsi dengan kunci publik si pengirim.




- Dengan cara ini, maka kerahasiaan pesan dan otentikasi si pengirim pesan keduanya dicapai sekaligus. Penerima pesan dapat mengotentikasi pengirim pesan karena kunci publik dan kunci privat adalah berpasangan. Penerima pesan juga dapat melakukan nir-penyangkalan jika pengirim menyangkal mengirim pesan
- Ide ini ditemukan oleh Diffie dan Hellman.

**Kesimpulan**: Jadi untuk menandatangani pesan, maka pesan dienkripsi dengan kunci privat si pengirim, penerima pesan mendekripsinya dengan kunci publik si pengirim pesan.




Sumber: https://www.educba.com/digital-signature-cryptography/

- Proses menandatantangani pesan (*signing* oleh pengirim):

  $S = E_{SK}(M)$

- Proses memverifikasi tanda-tangan (*verification* oleh penerima):

  $M = D_{PK}(S)$

  Keterangan:

  *SK* = *secret key* = kunci privat pengirim

  *PK* = *public key* = kunci publik pengirim

  *E* = fungsi enkripsi; *D* = fungsi dekripsi

  *M* = pesan; *S* = *signature* = cipherteks (hasil enkripsi pesan)

- Jadi, mengenkripsi pesan dengan menggunakan kunci privat sama artinya dengan menandatangani pesan. Mendekripsi pesan dengan kunci publik sama artinya dengan memverifikasi tanda-tangan .

- Dengan cara seperti ini, tidak lagi dibutuhkan pihak penengah (arbitrase).

- Beberapa algoritma kunci-publik dapat digunakan untuk menandatangani pesan dengan cara mengenkripsinya, asalkan algoritma tersebut memenuhi sifat:

$$D_{SK}(E_{PK}(M)) = M \text{ dan } D_{PK}(E_{SK}(M)) = M ,$$

Keterangan:

*PK* = kunci publik ; *SK* = kunci privat (*secret key*).
*E* = fungsi enkripsi; *D* = fungsi dekripsi; *M* = pesan

- Contoh algoritma yang memenuhi sifat ini adalah RSA, karena persamaan enkripsi dan dekripsi identik (dapat dipertukarkan)

Enkripsi/dekripsi biasa dengan RSA:
- Enkripsi: $c = m^e \bmod n$
- Dekripsi: $m = c^d \bmod n$

Tanda-tangan digital dengan RSA:
- Signing: $c = m^d \bmod n$
- Verification: $m = c^e \bmod n$

# Tanda-tangan Digital Menggunakan Kombinasi Kriptografi kunci-publik dan Fungsi *Hash*

- Menandatangani pesan dengan cara mengenkripsinya menggunakan kriptografi kunci-publik selalu memberikan dua fungsi berbeda: (1) kerahasiaan pesan dan (2) otentikasi pesan dan pengirim.

- Pada beberapa kasus, seringkali otentikasi yang diperlukan, tetapi kerahasiaan pesan tidak perlu. Maksudnya, pesan tidak perlu dienkripsi karena tidak rahasia, sebab yang dibutuhkan hanya keotentikan pesan dan pengirim saja.

- Kombinasi algoritma kunci-publik dan fungsi *hash* dapat digunakan untuk kasus seperti ini.

Alice

Signer

Verifier

Bob

Message

Fungsi
*Hash*

Message
Digest (*MD*)

Private
Key Alice

Encrypt

Signature

Message

Signature

Public
Key Alice

Decrypt

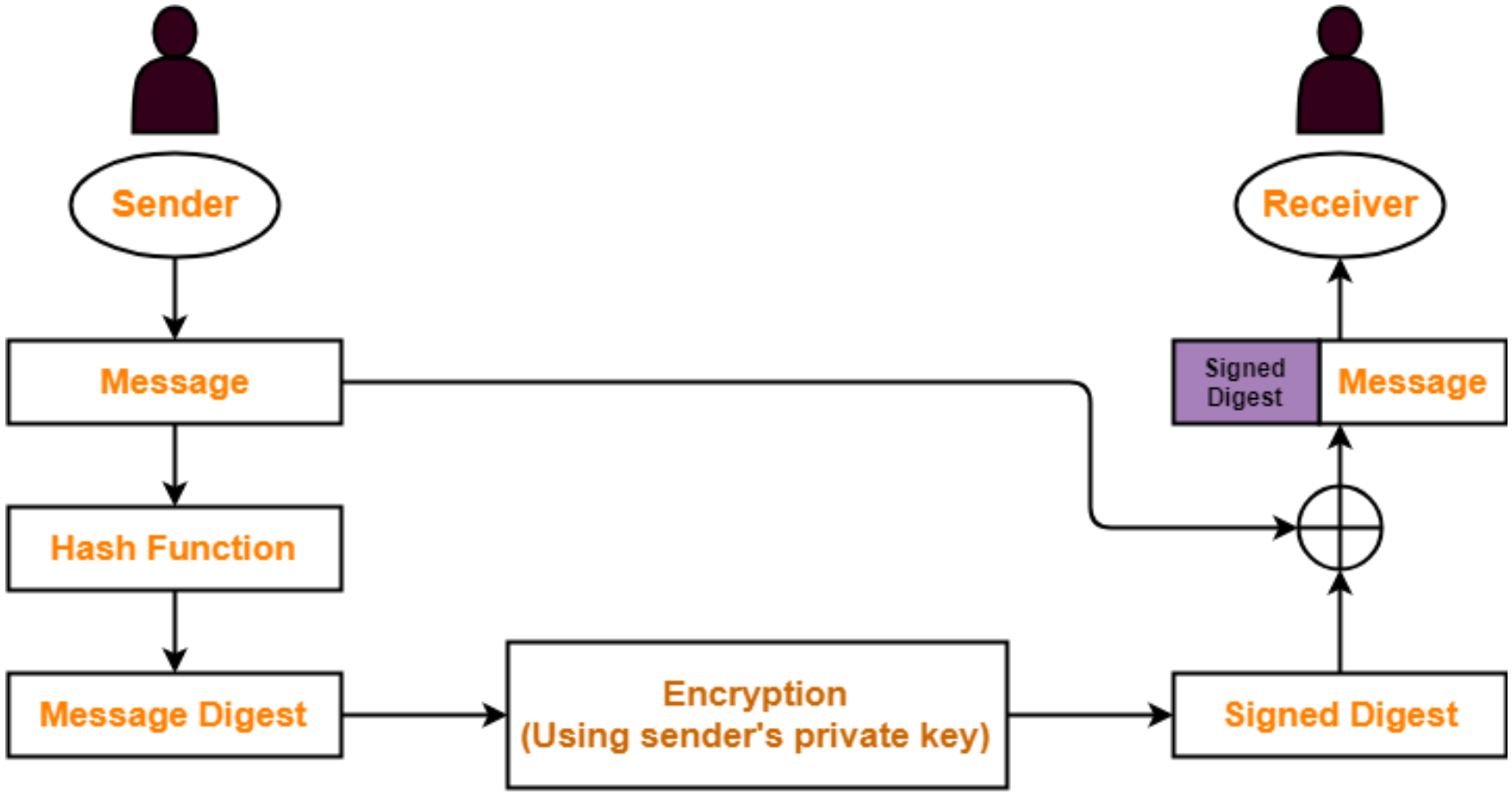
Message
Digest (*MD*)

?
=

Message

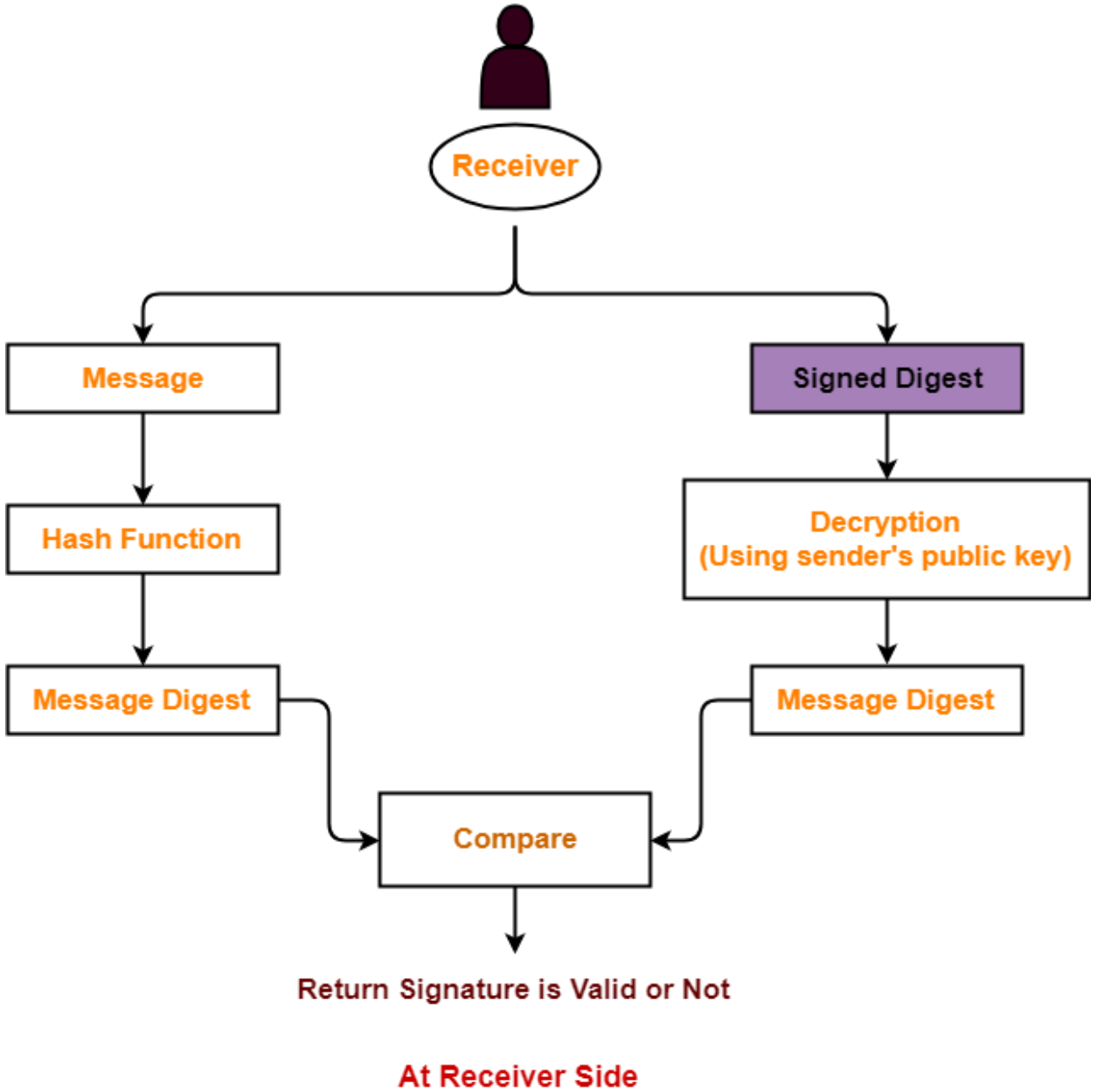
Fungsi
*Hash*

Message
Digest (*MD'*)




At Sender Side

https://www.gatevidyalay.com/how-digital-signature-works-algorithm/




https://www.gatevidyalay.com/how-digital-signature-works-algorithm/

- Dua algoritma *signature* yang digunakan secara luas adalah *RSA* dan *ElGamal Signature*.

- Pada *RSA*, algoritma enkripsi dan dekripsi identik, sehingga proses *signature* dan verifikasi juga identik.

- Sedangkan pada algoritma ElGamal, algoritma enkripsi dengan algoritma signatute tidak sama.

- Selain *RSA*, terdapat algoritma yang dikhususkan untuk tanda-tangan digital, yaitu *Digital Signature Algorithm* (DSA), yang merupakan bakuan (*standard*) untuk *Digital Dignature Standard* (DSS).

- Pada *DSA*, algoritma *signature* dan verifikasi berbeda

# Tanda-tangan digital dengan algoritma RSA

**Langkah-langkah pemberian tanda-tangan (*signing*)**

1. Pengirim menghitung nilai *hash* dari pesan *M:*

   $h = H(M)$

2. Pengirim mengenkripsi *h* dengan kunci privatnya (SK) menggunakan persamaan enkripsi *RSA,* hasilnya adalah signature *S*:

   $S = h^{SK} \bmod n$ (*n* adalah modulus, $n = pq$).

3. Pengirim mentransmisikan $M + S$ ke penerima

**Langkah-langkah verifikasi tanda-tangan (*verifying*)**

1. Penerima menghitung nilai *hash* dari pesan *M* yang diterima:

   $h = H(M)$

2. Penerima melakukan dekripsi terhadap tanda-tangan *S* dengan kunci publik si pengirim (PK) menggunakan persamaan dekripsi *RSA*:

   $h' = S^{PK} \bmod n$

3. Penerima membandingkan *h* dengan *h'*. Jika *h* = *h'* maka tanda-tangan digital adalah otentik. Jika tidak sama, maka tanda-tangan tidak otentik sehingga pesan dianggap tidak asli lagi atau pengirimnya


$M$ =

Pada wisuda sarjana baru ITB, ternyata ada seorang wisudawan yang paling muda. Umurnya baru 19 tahun. Ini berarti dia masuk ITB pada umur 15 tahun. Zaman sekarang banyak sarjana masih berusia muda belia. Mungkin masuk sekolah pada usia dini dan mengikuti kelas akselerasi pada tingkatan SD, SMP, dan SMA. Masuk SD umur 6 tahun dan ikut kelas aksel sehingga selesai dalam waktu lima tahun pada umur 11. SMP diselesaikan dalam waktu dua tahun dan SMA dalam waktu dua tahun, sehingga lulus SMA pada umur 15 tahun. Kuliah di ITB selama empat tahun sehingga wajar saja menjadi sarjana pada umur 19 tahun.

$h = H(M)$ = A4C05176E1440FC879C06C72FA603A24 (heksadesimal)

= 218991964599382371228554013295471770148 (desimal

$S = h^{PrivK} \bmod n$ ($n$ = 223427, $PrivK$ = 171635)

= $(218991964599382371228554013295471770148)^{171635} \bmod (223427) = 46489$

$M + S$ =

Pada wisuda sarjana baru ITB, ternyata ada seorang wisudawan yang paling muda. Umurnya baru 19 tahun. Ini berarti dia masuk ITB pada umur 15 tahun. Zaman sekarang banyak sarjana masih berusia muda belia. Mungkin masuk sekolah pada usia dini dan mengikuti kelas akselerasi pada tingkatan SD, SMP, dan SMA. Masuk SD umur 6 tahun dan ikut kelas aksel sehingga selesai dalam waktu lima tahun pada umur 11. SMP diselesaikan dalam waktu dua tahun dan SMA dalam waktu dua tahun, sehingga lulus SMA pada umur 15 tahun. Kuliah di ITB selama empat tahun sehingga wajar saja menjadi sarjana pada umur 19 tahun.

46489

Bob

| Pada wisuda sarjana baru ITB, ternyata ada seorang wisudawan yang paling muda. Umurnya baru 19 tahun. Ini berarti dia masuk ITB pada umur 15 tahun. Zaman sekarang banyak sarjana masih berusia muda belia. Mungkin masuk sekolah pada usia dini dan mengikuti kelas akselerasi pada tingkatan SD, SMP, dan SMA. Masuk SD umur 6 tahun dan ikut kelas aksel sehingga selesai dalam waktu lima tahun pada umur 11. SMP diselesaikan dalam waktu dua tahun dan SMA dalam waktu dua tahun, sehingga lulus SMA pada umur 15 tahun. Kuliah di ITB selama empat tahun sehingga wajar saja menjadi sarjana pada umur 19 tahun. |
|---|
| 46489 |

$h = H(M) = \text{A4C05176E1440FC879C06C72FA603A24}$ (heksadesimal)

$= 218991964599382371228554013295471770148$ (decimal)

$\equiv 125468 \pmod{223427}$

$h' = S^{PubK} \bmod n \quad (PubK = 731)$

$= (46489)^{731} \bmod (223427)$

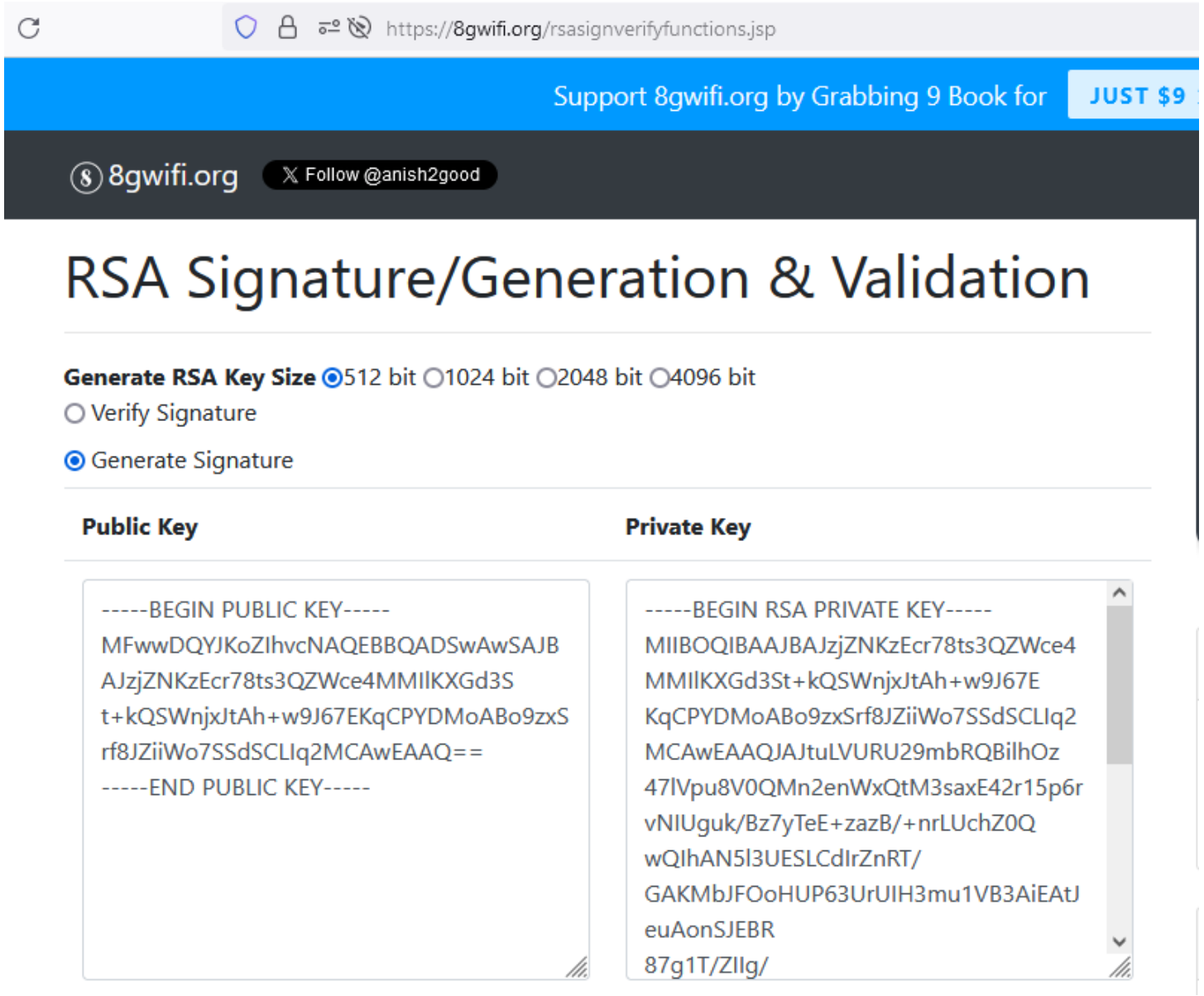
$= 125468$

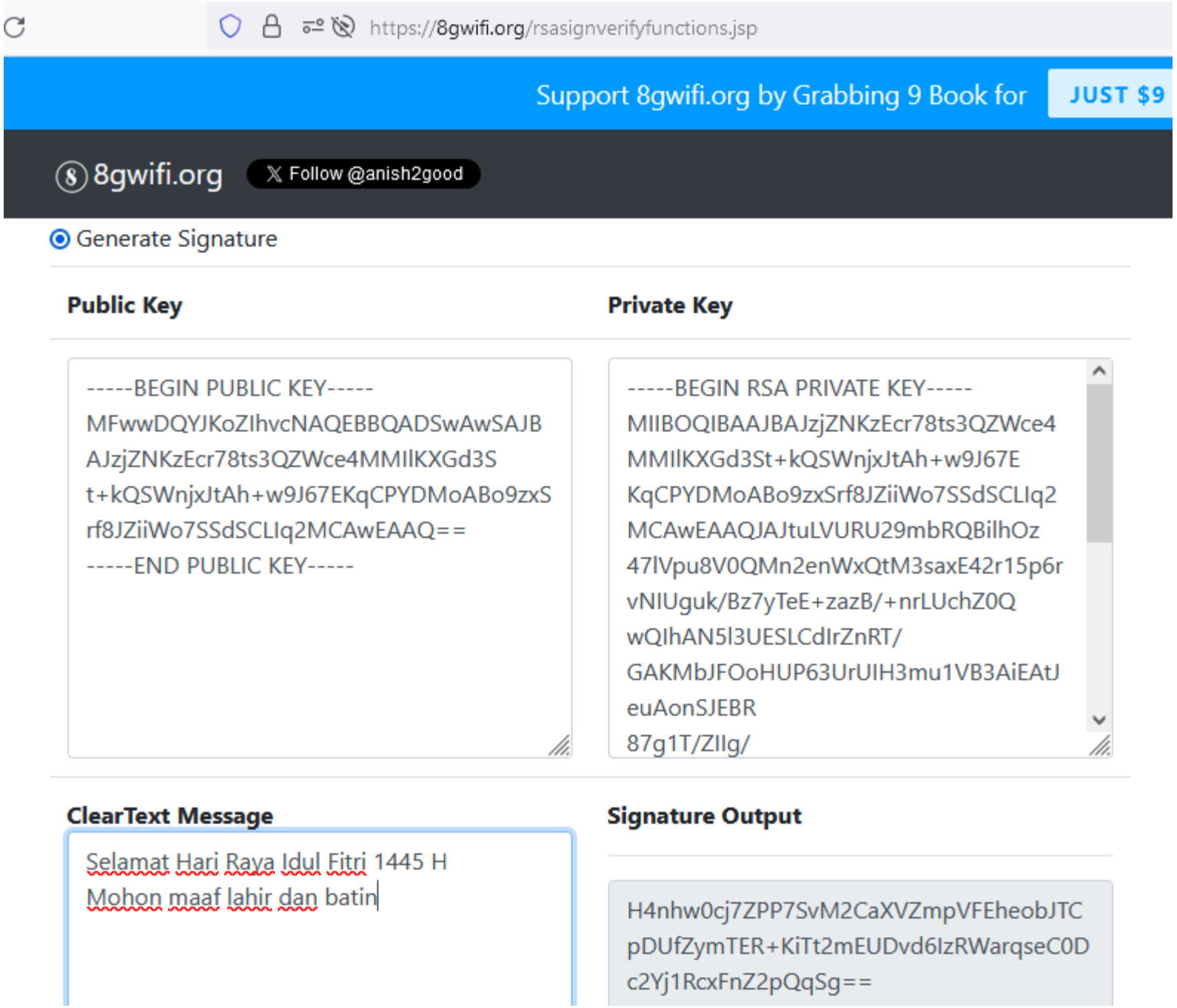
sama

Tandda tangan valid!

# Demo calculator digital signature online

https://8gwifi.org/rsasignverifyfunctions.jsp




https://8gwifi.org/rsasignverifyfunctions.jsp
Support 8gwifi.org by Grabbing 9 Book for
JUST $9
8gwifi.org
Follow @anish2good
Generate Signature
Public Key
-----BEGIN PUBLIC KEY-----
MFwwDQYJKoZIhvcNAQEBBQADSwAwSAJB
AJzjZNKzEcr78ts3QZWce4MMIlKXGd3S
t+kQSWnjxJtAh+w9J67EKqCPYDMoABo9zxS
rf8JZiiWo7SSdSCLIq2MCAwEAAQ==
-----END PUBLIC KEY-----
Private Key
-----BEGIN RSA PRIVATE KEY-----
MIIBOQIBAAJBAJzjZNKzEcr78ts3QZWce4
MMIlKXGd3St+kQSWnjxJtAh+w9J67E
KqCPYDMoABo9zxSrf8JZiiWo7SSdSCLIq2
MCAwEAAQJAJtuLVURU29mbRQBilhOz
47lVpu8V0QMn2enWxQtM3saxE42r15p6r
vNIUguk/Bz7yTeE+zazB/+nrLUchZ0Q
wQIhAN5l3UESLCdIrZnRT/
GAKMbJFOoHUP63UrUIH3mu1VB3AiEAtJ
euAonSJEBR
87g1T/ZIIg/
ClearText Message
Selamat Hari Raya Idul Fitri 1445 H
Mohon maaf lahir dan batin
Signature Output
H4nhw0cj7ZPP7SvM2CaXVZmpVFEheobJTC
pDUfZymTER+KiTt2mEUDvd6IzRWarqseC0D
c2Yj1RcxFnZ2pQqSg==


https://8gwifi.org/rsasignverifyfunctions.jsp
Support 8gwifi.org by Grabbing 9 Book for
8gwifi.org
Follow @anish2good
ClearText Message
Selamat Hari Raya Idul Fitri 1445 H
Mohon maaf lahir dan batin
Signature Output
Signature Verification Passed
Provide Signature Value (Base64)
H4nhw0cj7ZPP7SvM2CaXVZmpVFEheobJTC
pDUfZymTER+KiTt2mEUDvd6IzRWarqseC0D
c2Yj1RcxFnZ2pQqSg==

1445 H diubah menjadi 1446 H

**ClearText Message**

Selamat Hari Raya Idul Fitri 1446 H
Mohon maaf lahir dan batin

**Signature Output**

Signature Verification Failed

Provide Signature Value (Base64)

H4nhw0cj7ZPP7SvM2CaXVZmpVFEheobJTC
pDUfZymTER+KiTt2mEUDvd6IzRWarqseC0D
c2Yj1RcxFnZ2pQqSg==

Sekarang akan kita buat kode program tanda-tangan digital!

```
from Crypto.PublicKey import RSA
from Crypto.Signature import pkcs1_15
from Crypto.Hash import SHA256

def PembangkitanKunciRSA(filekuncipublik, filekunciprivat):
    # Buat kunci RSA
    key = RSA.generate(1024)
    kunci_publik = key.publickey()

    # Tampilkan informasi kunci
    print("Modulus (n):", key.n)
    print("Kunci publik (e):", key.e)
    print("Kunci privat (d):", key.d)

    #Simpan kunci publik dan kuncoi privat ke dalam dua file kunci berbeda dengan format PEM
    with open(filekuncipublik, "wb") as file:
        file.write(kunci_publik.exportKey('PEM'))
        file.close()

    with open(filekunciprivat, "wb") as file:
        file.write(key.exportKey('PEM','passwordku'))  # Password bisa diubah
        file.close()
```

```
def menandatangani(pesan, file_kunci):
    #Mendandatangni pesan dengan menggunakan kunci privat pengirim pesan

    # 1. Hitung nilai hash pesan
    pesan = pesan.encode('utf-8')
    nilai_hash = SHA256.new(pesan)

    # 2. Load kunci privat
    with open(file_kunci, "rb") as file:
        kunci_privat = RSA.import_key(file.read(), passphrase='passwordku')

    # 3. Buat tanda tangan digital dengan fungsi sign
    tanda_tangan = pkcs1_15.new(kunci_privat).sign(nilai_hash)

    return tanda_tangan
```

```
def memverifikasi(tandatangan, pesan, file_kunci):
    #Verifikasi tanda-tangan digital dengan menggunakan kunci publik pengirim pesan

    # 1. Hitung nilai hash pesan
    pesan = pesan.encode('utf-8')
    nilai_hash = SHA256.new(pesan)

    # 2. Load kunci publik
    with open(file_kunci, "rb") as file:
        kunci_publik = RSA.import_key(file.read())

    # 3. Verifikasi tanda tangan digital dengan fungsi verify
    try:
        pkcs1_15.new(kunci_publik).verify(nilai_hash, tandatangan)
        return 1  # Tanda tangan valid
    except (ValueError, TypeError):
        return 0  # Tanda tangan tidak valid
```

```
#Program utama

#Alice membangkitkan kunci publik dan kunci privatnya
print("\nKUNCI PUBLIK DAN KUNCI PRIVAT ALICE:")
PembangkitanKunciRSA('kunci_publik_Alice.pem', 'kunci_privat_Alice.pem')

#Alice mengetikkan pesan kepada Bob, lalu menandatangi pesan tersebut menggunakan kunci
privatnya (kunci privat Alice)
pesan = input("\nAlice, ketikkan pesan teks buat Bob: ")
tandatangandigital = menandatangani(pesan, 'kunci_privat_Alice.pem')

print("\nPesan: ", pesan)
print("\nTandatangan digital:", tandatangandigital.hex())

# Alice mengirim pesan + tandantangan digital kepada Bob

# Bob memverifikasi tanda-tangan di dalam pesan dengan menggunakan kunci publik Alice
hasil_verifikasi = memverifikasi(tandatangandigital, pesan, 'kunci_publik_Alice.pem')
if hasil_verifikasi == 1:
   print("\nTanda-tangan digital valid!")
else:
   print("\nTanda-tangan digital tidak valid!")
```

```
#Contoh hasil verifikasi yang salah
print("\nContoh pesan yang sudah diubah:")
#Misal pesan asli diubah, misalnya huruf pertama diganti dengan karakter *
pesan = '*' + pesan[1:]
print(pesan)
hasil_verifikasi = memverifikasi(tandatangandigital, pesan, 'kunci_publik_Alice.pem’)

if hasil_verifikasi == 1:
   print("Tanda-tangan digital valid!")
else:
   print("Tanda-tangan digital tidak valid!")
```

## Hasil run program:

KUNCI PUBLIK DAN KUNCI PRIVAT ALICE:
Modulus (n):
14321240605607385012642161934394110585703093013078659208243430994295661189215885137948326958485921271308763629016213249964405438166852787946627192312084687325432531877892806028148161452325775717460524381083368472323221000847769833446788700846719194862228079007146846718805097188175934328927508879531056249858 9
Kunci publik (e): 65537
Kunci privat (d):
27981672330866924803833389317472051825675283585222428729047276177114598093887332223236999970003543323482722472733358357232435362577864404787610235066640888435246404348647347556181013863331572022109952714357668197019390251667059094924634435720976564256142587558734101094125314045996842780688367666268087389313

Alice, ketikkan pesan teks buat Bob:  Guys, apakah kalian sudah membuat proposal Tugas Akhir?

Pesan:  Guys, apakah kalian sudah membuat proposal Tugas Akhir?

Tandatangan digital:
34ba0e6ee8d42f598d604532af0ee8297271cb1337c3dc08b35bb88be7667eff8eb9eec5783338bd578ce92bfd35bcbe75536dfa9855714a03cdb2467a1f3a9c8125e868a77b9b43f505e30e17a38c9510705f605f8e6323e16531df89e82da39dba6ed4c6d80212bfbdb219c1b22bbf8f3a11eaad77515293a9ceed65a5e206

Tanda-tangan digital valid!

Contoh pesan yang sudah diubah:
*uys, apakah kalian sudah membuat proposal Tugas Akhir?
Tanda-tangan digital tidak valid!

# SELAMAT BELAJAR