# SIGNCRYPTION DENGAN MENGGUNAKAN ELLIPTICAL CURVE CRYPTOGRAPHY

TUGAS MAKALAH
KEAMANAN & INFORMASI JARINGAN
(EL5241)

OLEH:
AGUS REZA ARISTIADI NURWA
23213307


SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
2015

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

# 1. *INTRODUCTION TO SIGNCRYPTION*

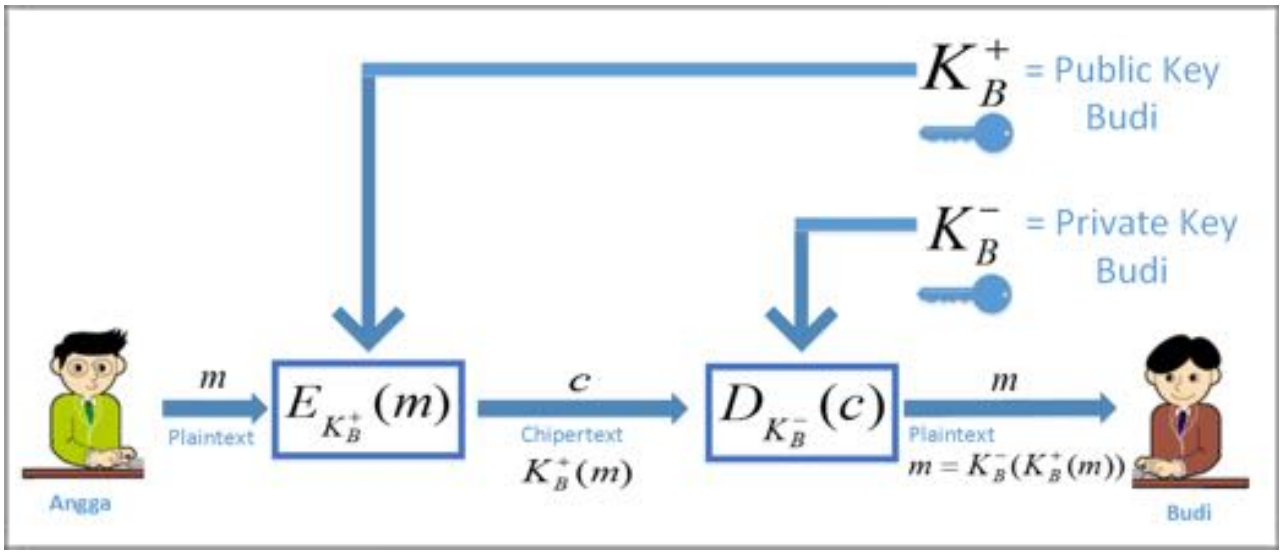
## 1.1 Asimetric Cryptography

Sebelum Asimetrik Kriptografi ditemukan, proses pengiriman pesan ataupun data secara *secure* antara kedua orang yang belum pernah bertemu merupakan hal mustahil. Karena diperlukan komunikasi pendahuluan diantara kedua orang tersebut untuk bertukar kunci rahasia.

Kemudian, ditemukanlah suatu metode yakni Asimetrik Kriptografi, metode ini yang kemudian digunakan secara luas sebagai metode pertukaran kunci Kriptografi Simetrik. Dengan menggunakan metode Kriptografi Asimetrik, proses komunikasi secara *secure* menggunakan jaringan *unsecure* diantara kedua orang yang belum pernah bertemu menjadi mungkin.

Asimetrik kriptografi merupakan sesuatu yang ajaib dalam bidang kriptografi. Dikatakan ajaib karena dalam asimetrik kriptografi, pesan dienkripsi dan didekripsi dengan kunci yang berbeda, tetapi kedua kunci yang berbeda tersebut saling berkaitan secara matematis dan berpasangan. Pertama kali di-eksplorasi oleh Diffie-Hellman, yang kemudian diimplementasikan secara praktikal oleh Rivest, Shamir dan Adleman. Nama ketiganya kemudian disematkan dalam nama algoritma Asimetrik Kriptografi yakni RSA. Algoritma RSA inilah yang kemudian sangat luas dipakai dalam transmisi data secara *secure*.

Asimetrik kriptografi juga disebut sebagai kunci Publik, untuk lebih jelasnya perhatikan contoh berikut yang direpresentasikan pada Gambar 1. Pada kasus ini, diasumsikan Angga dan Budi belum pernah bertemu satu sama lainnya, tetapi Angga ingin mengirim pesan rahasia kepada Budi. Diasumsikan juga, jalur komunikasi yang akan dipakai Angga dan Budi tidak diketahui tingkat keamanannya.




**GAMBAR.1: METODE ASIMETRIK KRIPTOGRAFI**

Angga mengirimkan pesan rahasia tersebut dengan menggunakan Asimetrik kriptografi. Syaratnya, Budi harus mempunyai sepasang kunci, yakni **Public Key** Budi dan **Private Key** Budi. **Public Key** milik Budi boleh diketahui siapapun. Sedangkan **Private Key**, hanya Budi seorang yang boleh mengetahuinya.

Prosesnya, yang pertama, Angga mengenkripsi pesan rahasia (**m**) dengan menggunakan **Public Key** milik Budi. Pesan yang telah dienkripsi ini disebut *ciphertext* atau **c**. *Ciphertext* ini tidak bisa didekripsi lagi dengan **Public Key** Budi. Kemudian proses kedua, ciphertext ini dikirimkan ke Budi dengan menggunakan jalur *unsecure*. Proses

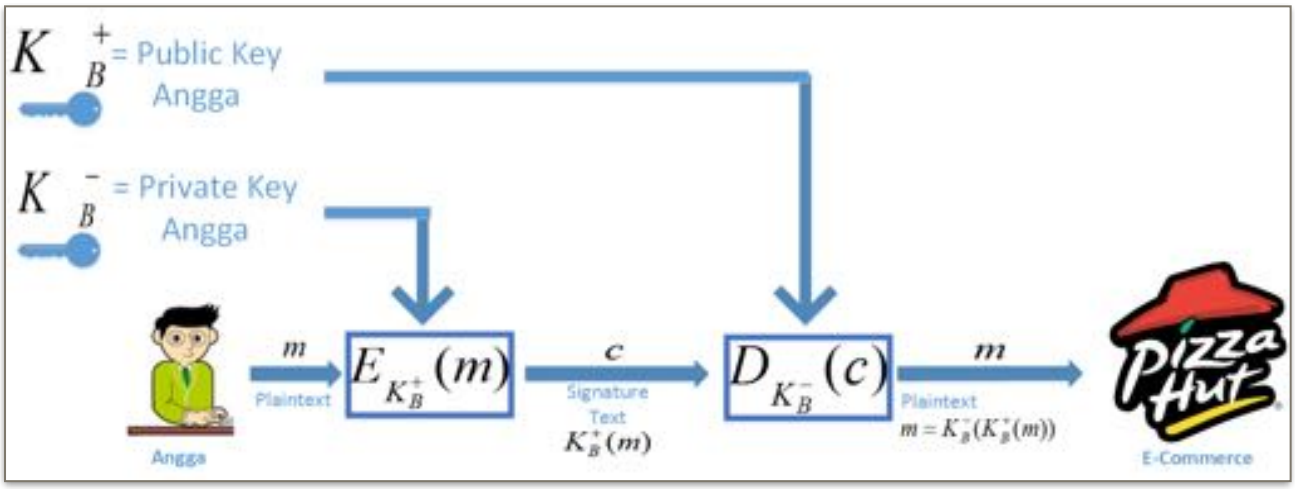
yang ketiga, Budi menerima ciphertext ini yang kemudian akan didekripsi menggunakan **Private Key** milik Budi. Pesan dari Angga pun dapat terbaca oleh Budi.

1.2 Digital Signature

Dari metode Asimetrik Kriptografi, akhirnya bisa juga dikembangkan menjadi metode untuk melakukan tanda tangan digital atau *digital signatures*. Tanda tangan digital bukan merupakan tanda tangan dengan menggunakan *balpoint* yang kemudian di-scan atau di-kuantisasi menjadi data digital.

Digital signatures adalah salah satu aplikasi dari asimetrik kriptografi yang digunakan untuk mensimulasikan ***security properties*** dari sebuah tanda tangan, hanya saja diaplikasikan secara digital [1]. Prinsip penggunaannya berbeda dari tanda tangan yang biasa dilakukan pada dokumen lembaran kertas yang dibubuhi tanda tangan.

Skema cara kerja digital signatures yang umum digunakan menggunakan dua buah algoritma. Algoritma pertama digunakan untuk penanda-tanganan (*signing)*, proses *signing* melibatkan kunci rahasia *author* atau ***private key***. Sedangkan algoritma kedua digunakan untuk melakukan verifikasi terhadap tanda tangan yang dilakukan *author*, proses verifikasi melibatkan kunci publik dari user atau ***public key***.



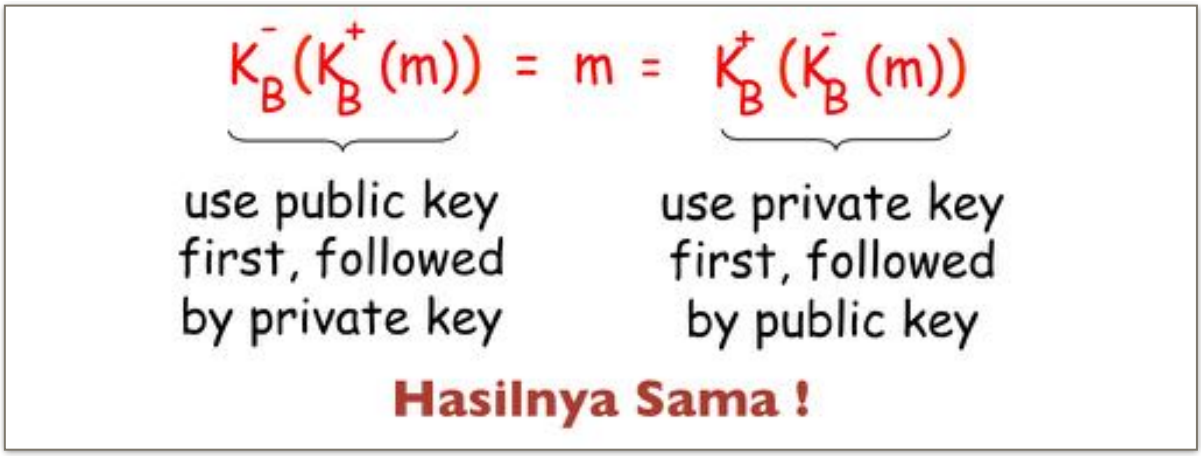

**GAMBAR.2: METODE DIGITAL SIGNATURE**

Contoh kasus penggunaan Digital Signature. Suatu hari Angga ingin melakukan pemesanan 10 loyang pizza ke restoran cepat saji, Angga mengirim pesan yang berisi informasi pemesanan tersebut secara online. **Pesan** yang akan dikirim (m) terlebih dahulu dilakukan fungsi enkripsi dengan menggunakan *Private Key* Angga sehingga menjadi **chipertext** (c). Ketika **chipertext** (c) ini sampai di Pizza Hut, kemudian **chipertext** ini dilakukan proses dekripsi dengan menggunakan *Public Key* Angga. Dengan begitu, setiap orang bisa membuka pesan yang berisi informasi pemesanan pizza Angga, karena *Public Key* Angga bisa diketahui bebas oleh setiap orang.

Digital Signature, hanya menjamin Integritas, Authentifikasi, dan *Non-Repudiation*. tapi tidak menjamin kerahasiaan pesan yang dikirim. Integritas pesan yang dikirim terjaga, sehingga pesan yang dikirim tidak bisa diubah oleh orang lain. Fungsi Authentifikasi, membuat pihak penerima bisa memastikan dengan pasti, sumber pesan berasal. Sedangkan *Non-Repudiation*, menjaga bahwa pihak pengirim tidak bisa menyangkal pesan yang telah dikirim, karena Private Key terikat hanya dimiliki oleh Angga. Sehingga pada contoh Angga memesan Pizza, ketika pesanan Pizza datang, Angga tidak dapat menyangkal pesanan tersebut bukan berasal dari dirinya.

Pada penggunaan pertama, Asimetrik Kriptografi dapat digunakan sebagai Algoritma Kriptografi berkirim pesan yang aman, sedangkan pada contoh kedua yakni

studi kasus pemesanan pizza, Asimetrik Kriptografi dapat digunakan sebagai Digital Signature. Keistimewaan ini didapat karena pada operasi matematika Asimetrik Kriptografi, penggunaan Publik Key lebih dahulu kemudian dibuka dengan menggunakan Private Key akan sama hasilnya dengan melakukan operasi Private Key terlebih dahulu baru kemudian dibuka dengan menggunakan Public Key.



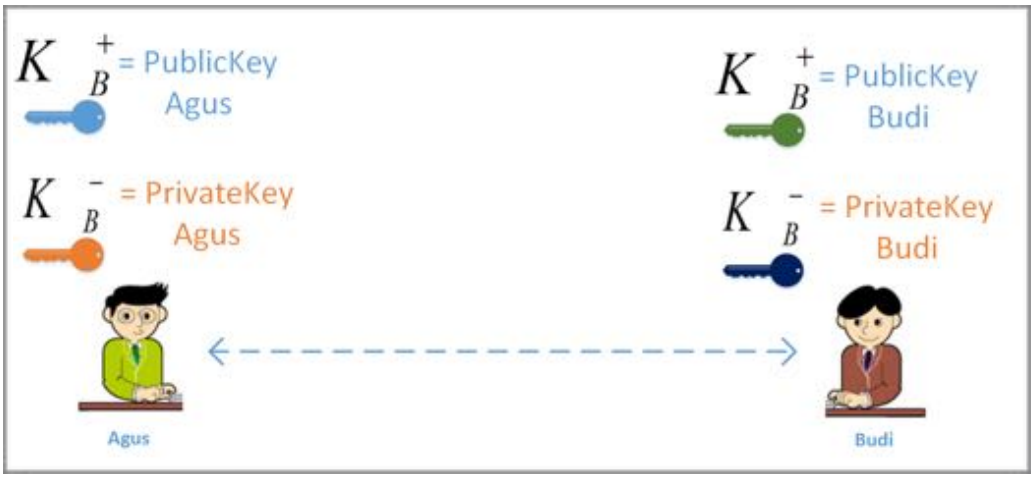

**GAMBAR.3: SIFAT MATEMATIS ASIMETRIK KRIPTOGRAFI**

1.3 *Signature then Encryption*

Sekarang pertanyaannya, bagaimana caranya agar pesan yang dikirim bisa di autentifikasi pengirimnya dan pesan tersebut dapat dikirim secara aman?. Mari perhatikan contoh kasus ketiga. Budi adalah seorang pengacara yang berada di Boyolali, sedangkan Agus adalah seorang klien pengacara tersebut yang berada di Jakarta. Suatu hari, Agus ingin mengirim draf dari suatu kontrak perusahaannya ke Budi.

Agus ingin memberikan jaminan kepada Budi bahwa kontrak yang dikirimkan tidak akan berubah selama dalam pengiriman ataupun dapat diubah oleh pihak ketiga sebelum sampai ke Budi dan Agus pun ingin memberikan jaminan bahwa yang dikirimkan adalah kontrak asli dari Agus.

Diasumsikan Agus akan mengirim kontrak tersebut melalui media Internet. Sehingga tidak diketahui keamanan media komunikasi yang dipakai antara Agus dan Budi, Agus pun berencana mengirimkan dokumen kontrak tersebut secara rahasia ke Budi agar tidak ada orang yang mengetahui isi dari kontrak tersebut.



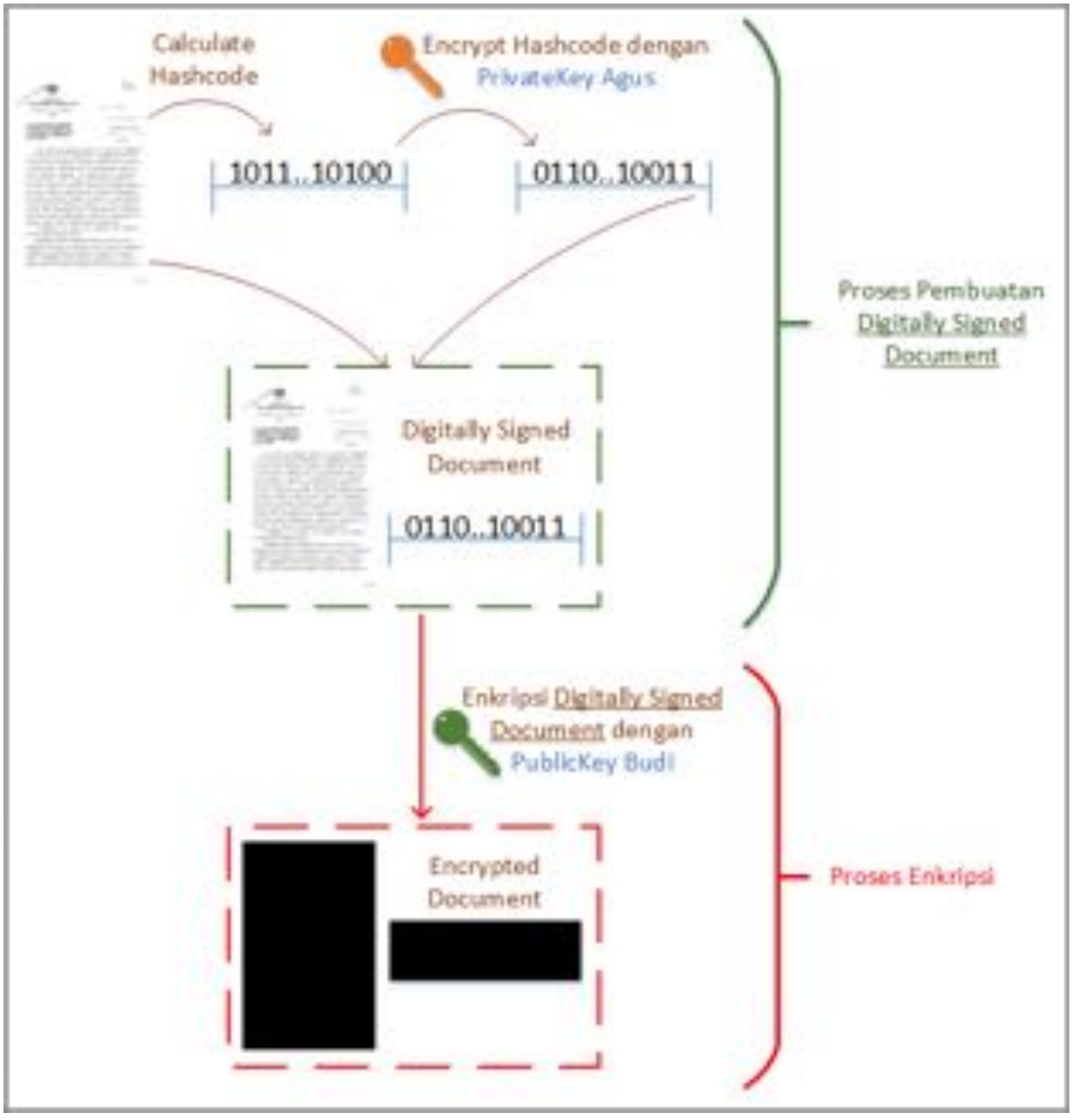

**GAMBAR.4: PROPERTIES KUNCI DARI KOMUNIKASI ASIMETRIK KRIPTOGRAFI.**

Jadi bagaimanakah cara Agus dan Budi agar bisa berkomunikasi secara aman dan bagaimanakah caranya agar Budi dapat memastikan integritas dokumen kontrak tersebut merupakan asli yang dikirimkan oleh Agus?. Permasalahan komunikasi seperti kasus

Agus dan Budi ini dapat dipecahkan dengan menggunakan metode *Signature-than-Encryption*.

Sebelum memulai berkomunikasi, kedua orang ini harus mempunyai sepasang kunci. Agus mempunyai sepasang kunci yakni **Public Key Agus** dan **Private Key Agus**. Sedangkan Budi juga mempunyai sepasang kunci yakni **Publik Key Budi** dan **Private Key Budi**.

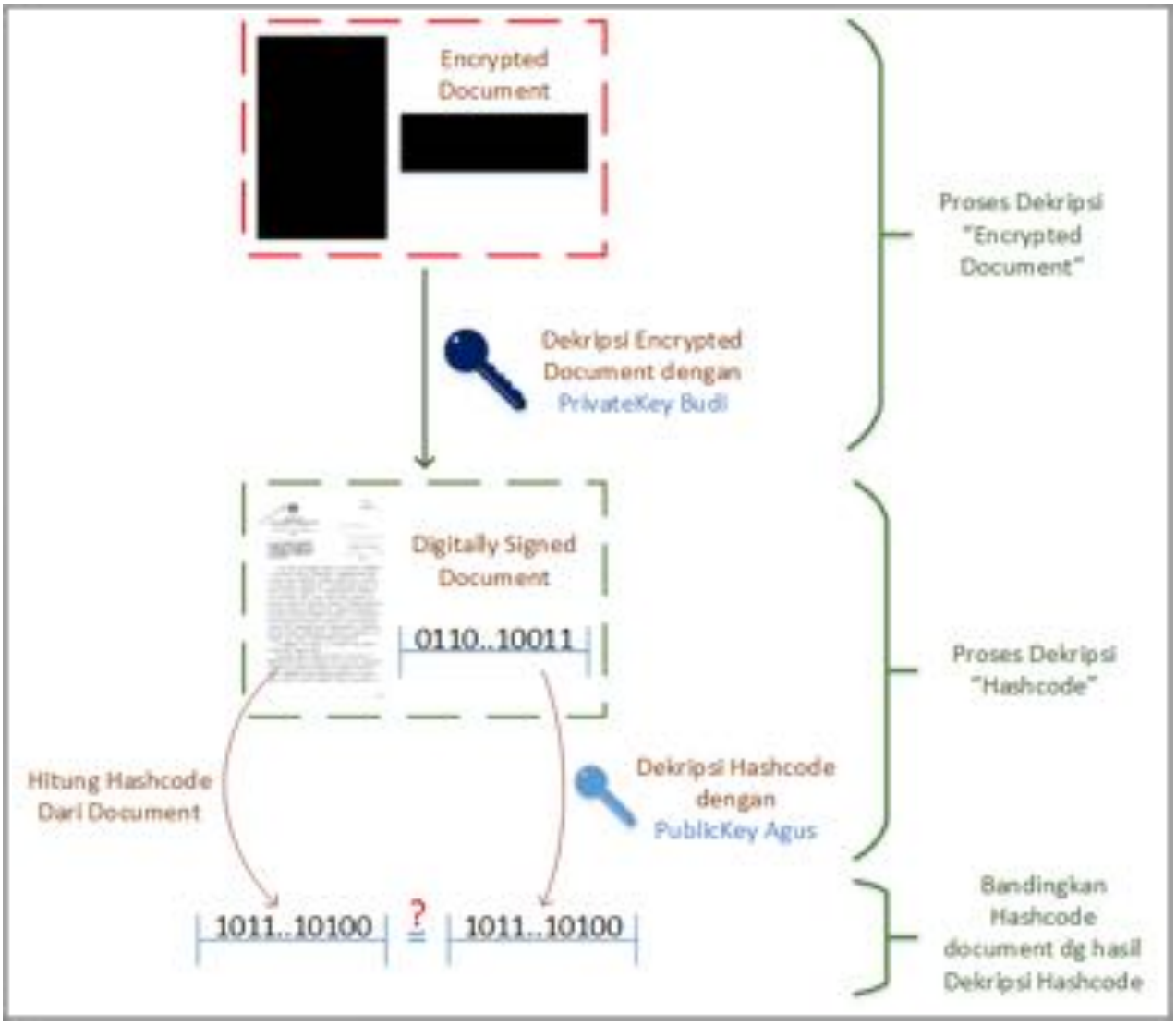
Proses pertama yang harus dilakukan, Agus harus menghitung *Hascode* dari dokumen kontrak yang akan dikirim, kemudian menanda tangani *Hashcode* tersebut dengan **Private Key Agus**. Kemudian dokumen dan hashcode tersebut ditempatkan di dalam satu **koper** atau *briefcase*. briefcase tersebut di enkripsi dengan menggunakan **Public Key Budi**. Pesan yang telah terenkripsi ini kemudian dikirimkan.




**GAMBAR.5: SIGNATURE THEN ENCRYPTION.**

Setelah pesan diterima oleh Budi, kemudian pesan tersebut di dekripsi dengan menggunakan Private Key Budi. Di dalam pesan tersebut terdapat dua komponen, yakni dokumen kontrak dan Hashcode dari dokumen kontrak tersebut yang ditanda tangani dengan menggunakan **Private Key Agus**. Terlebih dahulu Hashcode tersebut dibuka

dengan menggunakan **Publik Key Agus.** Pada dokumen yang diterima juga dilakukan hashcode ulang, yang kemudian dibandingkan antara hashcode ulang dengan nilai hashcode yang telah dibuka dengan **Publik Key Agus**. Bila hashcode tersebut bernilai sama, maka dapat dipastikan dokumen tersebut adalah asli yang dikirimkan oleh Agus.




**GAMBAR 6: PROSES DEKRIPSI SIGNATURE TERENKRIPSI.**

1.4 *Signcryption*

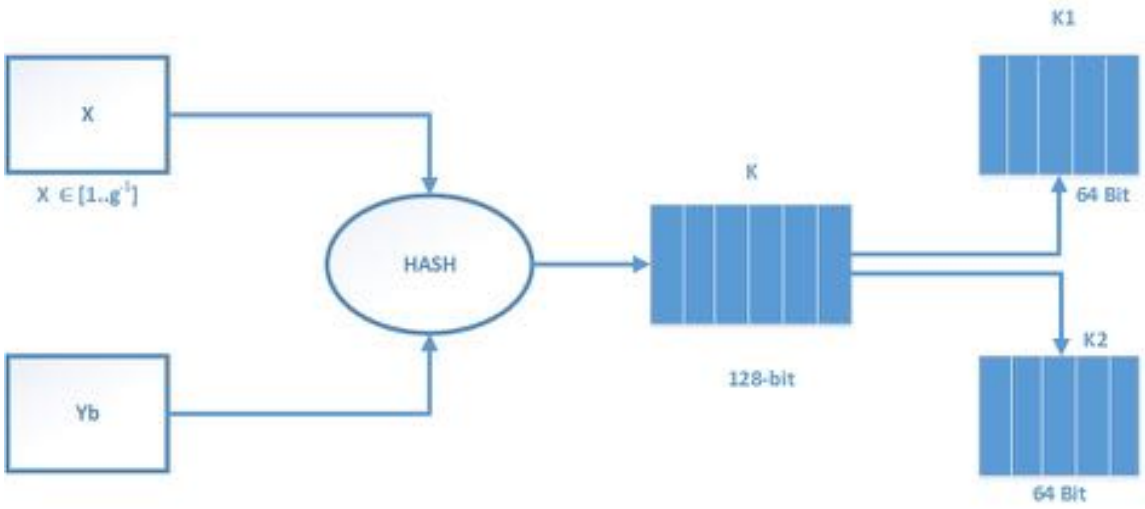
Pada metode sebelumnya, diperkenalkan metode *signature then encryption*, metode tersebut sangat cukup untuk memenuhi persyaratan keamanan, autentifikasi serta *non-repudiation*. Tetapi ada masalah yang mendasari, yakni

a. Membutuhkan banyak *machine cycle,* pada pengiriman saja terdapat dua kali proses, yakni Sign dan Encrypt.

b. Konsep kriptografi Publik membutuhkan besar kunci bit berkisar antara 1024 sampai 4096 bit, sehingga setiap proses membutuhkan banyak *resources* processor. Satu proses saja membutuhkan banyak resources processor, apalagi jika dilakukan dua kali secara berurutan.

c. Machine Cycle dan besarnya bit kunci Asimetrik Kriptografi berimbas pada waktu yang dibutuhkan untuk memverifikasi signature dan dekripsi.

d. Terjadi penambahan bit-bit pada pesan awal.




**GAMBAR 7: PEMBENTUKAN KEY BARU DARI HASH PRIVATE KEY DAN PUBLIC KEY**

e. Usaha yang dibutuhkan untuk mengirimkan pesan adalah penjumlahan antara usaha untuk melakukan digital signature dan usaha untuk melakukan enkripsi, dengan kata lain boros sumber daya.

Pertanyaannya adalah apakah mungkin untuk mengirimkan sebuah pesan yang mempunyai panjang tak beraturan, hanya dengan membutuhkan sumber daya yang lebih sedikit dari metode signature then encyption. Ternyata pada tahun 1997, Yuliang Zheng dari Monash University di Australia telah menjawab pertanyaan tersebut dengan menemukan *cryptography primitive* baru yang disebut *signcryption*.

Jadi, signcryption adalah paradigma baru dalam kriptografi dengan menggunakan kunci publik, dimana metode baru ini secara bersamaan memenuhi fungsi *digital signature* dan *publik key encryption* dalam satu langkah pengerjaan. Karena satu langkah pengerjaan, maka berdampak signifikan pada sedikitnya penggunaan sumber daya processor dibandingkan metode tradisional *signature then encryption* [8].

*Signcryption* harus memenuhi *correctness*, *security*, dan efisiensi [4]. *Corretcness* pesan yang dienkripsi dapat diverifikasi kebenarannya. Efisiensi adalah waktu komputasi yang dibutuhkan signcryption harus lebih pendek dibandingkan waktu komputasi metode sign then encrypt. *Security* dari signcryption dibuat dengan mempertimbangkan hal-hal berikut [3]:

1. Integritas. Pesan yang diterima benar-benar pesan yang asli dari pengirim yang bersangkutan tanpa adanya perubahan saat ditransmisikan.

2. Autentikasi. Pesan, dokumen, data, atau informasi yang dikirimkan dapat diverifikasi identitas pengirimnya.

3. *Non-repudiation*. Sumber pengirim tidak dapat menyangkal asal-usul pesan, dokumen, data, atau informasi yang telah dikirimkan sebelumnya.

4. *Forward secrecy*. Ketidaklengkapan informasi yang didapatkan penyadap pesan menjadikan pesan tetap terjaga kerahasiaannya. Contohnya meskipun penyadap telah mendapatkan private key pengirim tidak menjadikan pesan yang dikirim dapat dibaca.

5. *Public verifiability*. Pihak ketiga yang memiliki wewenang dapat memverifikasi pesan yang dikirimkan adalah valid atau tidak valid tanpa membutuhkan *private key* pengirim ataupun *private key* penerima.

6. *Un-forge-ability*. Tidak mungkin bagi siapapun untuk membuat *private key* dan *public ke*y palsu menggunakan identitas entitas lain sehingga dapat membuat pesan seakan-akan dari entitas lain.

7. Kerahasiaan. Meskipun semua bagian pesan yang dikirimkan (c, r, s) mampu didapatkan oleh penyadap. Namun, selama penyadap tidak memiliki *private key* penerima, p, dan q maka pesan tidak dapat dibaca oleh penyadap.

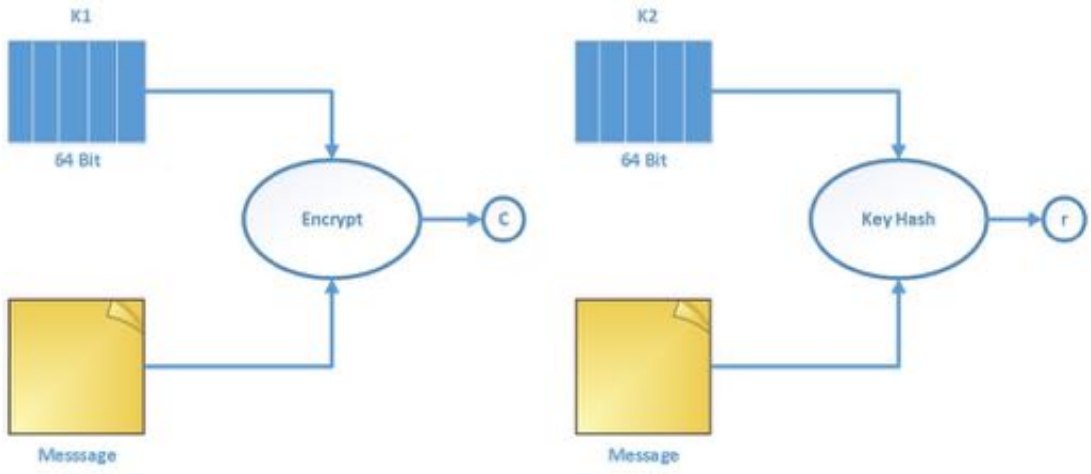
Bagaimana cara kerja Signcryption?

Langkah 1. Signcrypting a message (di Angga)

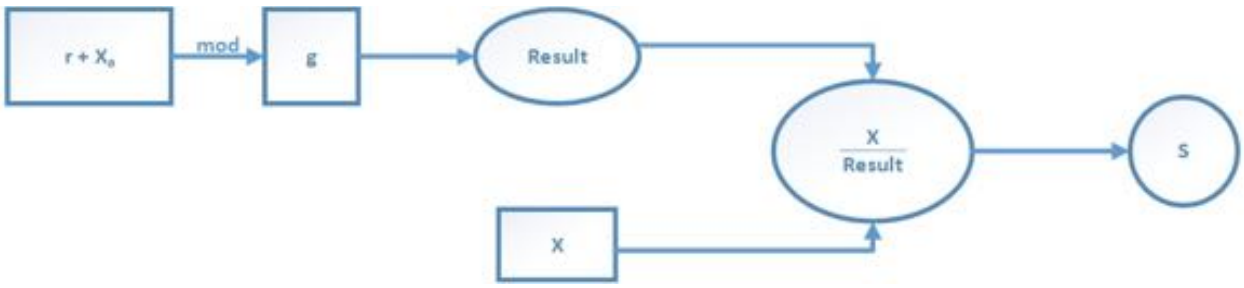
1. Pilih satuan nilai x, dari large range 1, …, q-1

2. Menggunakan kunci publik Budi dan dengan nilai x, dan menghitung hash dari nilai x dan nilai publikBudi. nilai x dan nilai publik Budi memberikan 128 bit string.

3. Membagi 128 bit nilai k, ke dalam 64 bit (k1,k2), kunci yang berpasangan.




**GAMBAR 8: PEMBENTUKAN CIPHER (C) DAN SIGNATURE (R)**




**GAMBAR 9: PEMBENTUKAN S SEBAGAI KEY PAIR SIGNATURE (R)**

4. Mengenkripsi pesan(m) menggunakan kunci publik skema enkripsi (E) dengan kunci k1, chiper text c = E.k1(m)

5. Menggunakan kunci k2 dalam satu arah, fungsi kunci hash KH untuk mendapatkan hash dari pesan(m) , dengan memanggil 128 bit

6. Menghitung nilai dari S dari :

- nilai dari x
- kunci pribadi Angga Xa

- nilai dari r

$$S = \frac{x}{(r + x_a) mod\ q}$$

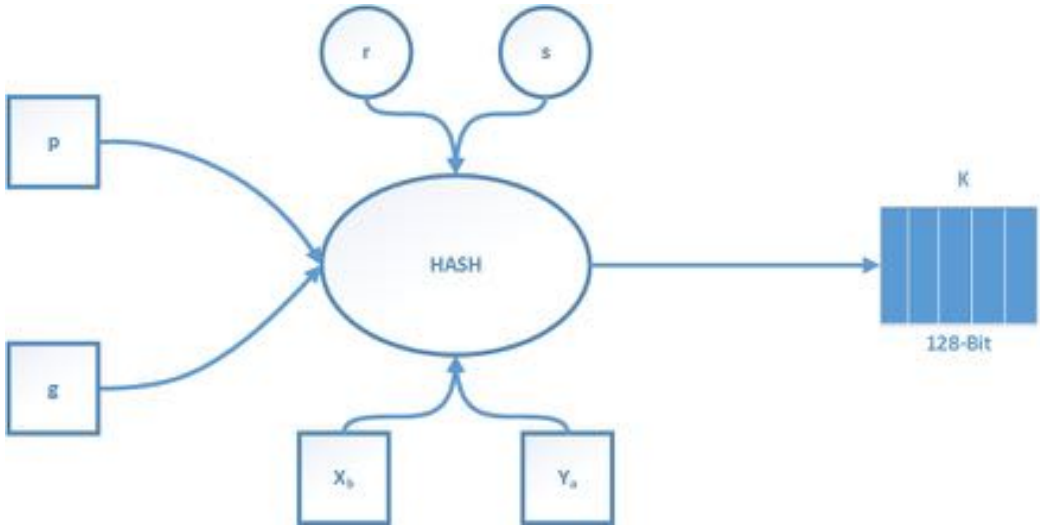
7. Sekarang Angga menpunyai 3 nilai yaitu: c,r,s
8. Kirim ke tiga nilai tersebut,


**GAMBAR 10: PENGIRIMAN PESAN YANG TELAH TER-SIGN DAN TER-ENCRYPT**

Langkah 2. Unsigncrypt message (di Budi)

1. Menerima ke tiga nilai yang dikirimkan Angga untuk Budi (c,s,r).
2. Untuk menghitung hash, Budi menggunakan nilai dari r dan s, kunci pribadi Budi Xb, kunci publik Angga Ya, P dan g
3. Hal diatas akan memberikan Budi hasil 128 bit

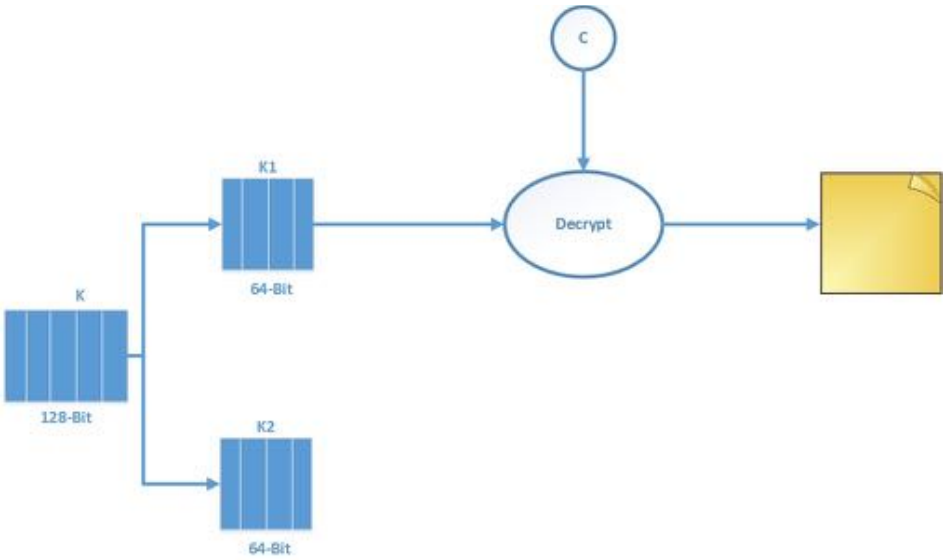
   k= hash (( Ya*g.r) s*Xb mod p)




**GAMBAR 11: PEMBENTUKAN KEY BARU DI SISI PENERIMA UNTUK DECRYPT DAN VERIFIKASI SIGNATURE**

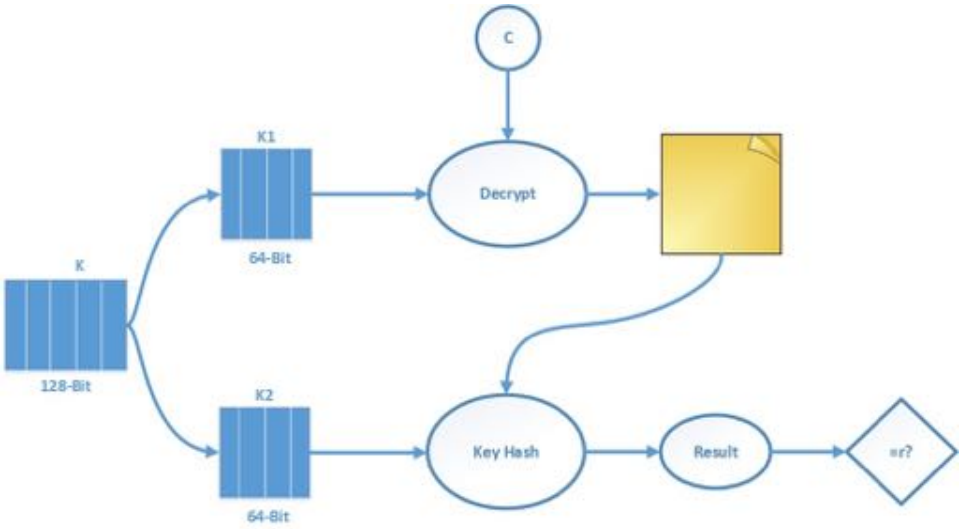
4. Hasil hash 128 bit dibagi menjadi dua, masing- masing 64 bit (k1,k2), kunci berpasangan.
5. Pasangan kunci (k1,k2) akan diidentifikasi ke kunci berpasangan yang akan dihasilkan ketika signcrypting the message

6. Budi menggunakan kunci k1 untuk mendeskripsikan chipertext c, yang akan memberi Budi pesan (m). m= D.K1(c)




**GAMBAR 12: DEKIPSI PESAN CIPHER**

7. Budi mengirimkan secara satu arah fungsi kunci hash (KH) pada m dengan menggunakan kunci k2 dan membandingkan hasil dengan nilai r yang diterima dari Angga
8. Jika cocok, pesan (m) akan ditandai dan dikirim oleh Angga
9. Jika tidak cocok, pesan (m) tidak akan ditandai oleh Angga atau akan disambut dan dimodifikasi oleh penyusup.
10. Budi menerima pesan (m) jika dan hanya jika KH.k2(m)= r




**GAMBAR 13: VERIFIKASI DIGITAL SIGNATURE PESAN YANG TELAH DI-DECRYPT**

## 2. INTRODUCTION TO ELLIPTICAL CURVE CRYPTOGRAPHY (ECC)

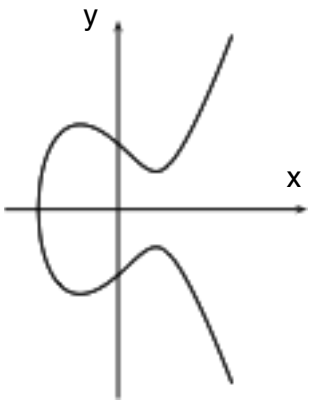
*Asymmetric cryptography* atau disebut juga *public key cryptography* diyakini sebagai enkripsi terbaik untuk saat ini [2]. Asimetrik kriptografi membutuhkan algoritma yang mudah diproses ke satu arah, tetapi sulit untuk mebalik hasil proses ke arah sebelumnya (*trapdoor function*) [2].

Setelah penemuan RSA dan metode Diffie-Hellman, peneliti mencari metode kriptografi lainnya selain menggunakan pemfaktoran hingga digunakannya salah satu cabang bahasan matematika yaitu *elliptic curves.* ECC merupakan menggunakan teori matematika *elliptic curve* yang diterapkan pada p*ublic key cryptography*. Public key cryptography memiliki komponen: *public key* dan *private key*. ECC pun nantinya akan memiliki komponen *public key* dan *private key*. ECC merupakan kumpulan titik (x, y) yang membentuk kurva solusi dari persamaan *elliptic curve* [6].

**TABEL 1: KOMPONEN ECC [5]**

| Konstanta | Private key | Public key | Operasi |
|---|---|---|---|
| G, a, b | nomor acak | Private key * G | y^2 = x^3 + ax + b<br>di mana 4a^3 + 27b^2 =/ = 0 |

Dari persamaan operasi tabel 1 disimpulkan bahwa *elliptic curve* (EC) merupakan kumpulan titik koordinat yang memenuhi persamaan dalam 2 variabel dengan masing-masing berderajat 2 dan 3. Dengan kata lain, nilai yang digunakan dalam ECC adalah titik yang berada di garis kurva.




**GAMBAR 14: KURVA ELLIPTIC CURVE (EC)**

Operasi grup pada EC adalah *point addition* dan *point doubling*. *Point addition* merupakan operasi menjumlahkan titik P (x1,y1) dengan titik Q (x2,y2) sehingga didapatkan titik baru R (x3,y3). P + Q = R. *Point doubling* merupakan cara pada EC untuk mendapatkan hasil kuadrat dari titik P. Untuk melakukan iterasi order P, nilai P dirubah menjadi bentuk biner terlebih dahulu. Dalam setiap iterasi ECC dibutuhkan *point doubling* untuk mencapai nilai iterasi biner P yang bertambah 1-bit tiap iterasi. Apabila ditemukan nilai hasil *doubling* yang belum memenuhi nilai biner p maka dilakukan *point addtion*. Berikut ini adalah persamaan untuk mendapatkan nilai (x3, y3) [6].

x3 = s^2 - x1 - x2 mod p

y3 = s(x1 - x3) - y1 mod p

Dalam persamaan sebelumnya dibutuhkan nilai s. Nilai s merupakan kemiringan dari garis pemotong kurva. Untuk point addtion dengan nilai P =/= Q maka nilai s didapatkan dari s = (y2 - y1) / (x2 - x1). Sedangkan untuk point doubling P = Q nilai s didapatkan dari s = (3x1^2 + a) / 2y1. Berikut ini adalah contoh penggunaan point addition dan point doubling dalam ECC.

Ambil contoh misalnya nilai a=2, b=2, p=17 didapatkan kurva y^2 = x^3 + 2x + 2 mod 17. Kemudian pada kurva tersebut diambil sebuah titik P = (5,1). Pada EC, nilai P + P contoh di bawah tidak boleh langsung dijumlahkan hasilnya menjadi (x3,y3) = (10,2). Namun, EC membutuhkan nilai s terlebih dahulu.

2P = P + P = (x1,y1) + (x2,y2) = (5,1) + (5,1) = (x3,y3)

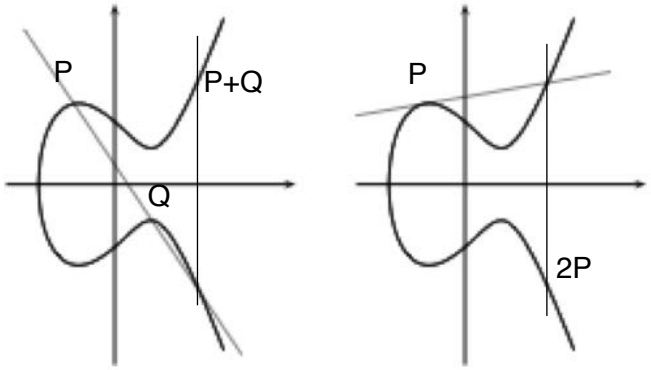
s = (3x1^2 + a) / 2y1 = (3*5^2 + 2) / (2*1) = 77 / 2

Perlu diingat bahwa nilai s di atas perlu disesuaikan dengan mod p (mod 17) dengan menggunakan Extended Eucledian Algorithm sehingga nilai s = 13 mod 17. Selanjutnya nilai x3 dan y3 dapat dihitung menggunakan nilai s.

x3 = s^2 - x1 - x2 = 13^2 -5 - 5 = 159

Sama seperti sebelumnya, nilai x3 disesuaikan dengan (mod 17) menjadi x3 = 6 mod 17. Kemudian nilai y3 dihitung menggunakan nilai s dan x3 yang telah didapatkan menggunakan perhitungan sebelumnya.

y3 = s(x1 - x3) -y1 = 13(5 - 6) - 1 = -14 = 3 mod 17.



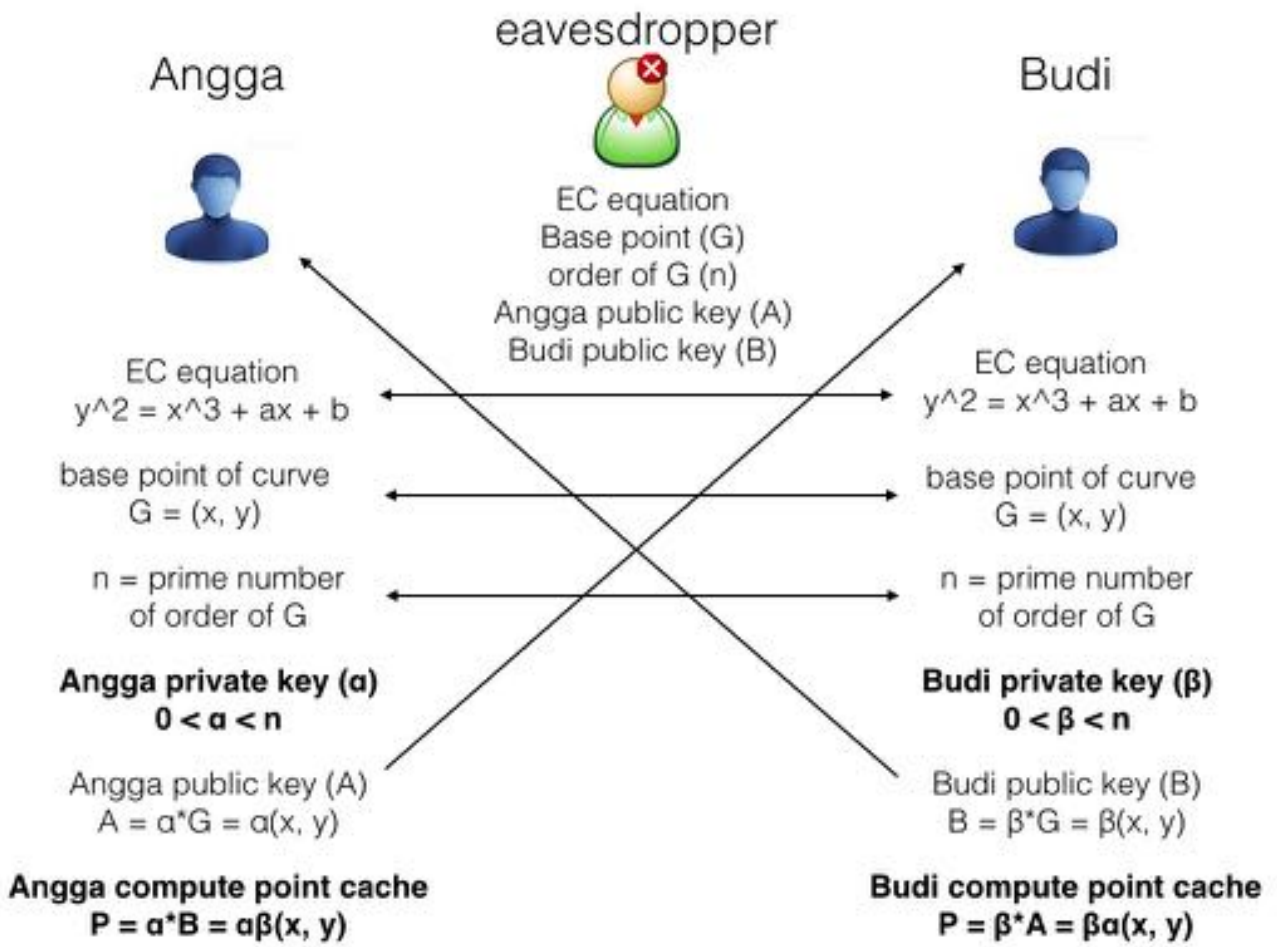

**GAMBAR 15: POINT ADDITION (KIRI) DAN POINT DOUBLING (KANAN) PADA KURVA ELLIPTIK (EC)**

Sekarang telah didapatkan nilai *point doubling* 2P = (6,3). Pertanyaan berikutnya, bagaimana jika ingin didapatkan nilai dari 3P? *Point addition* dilakukan untuk mendapatkan 3P dengan cara 3P = 2P + P. Pencarian nilai s berbeda antara *point doubling* yang telah dilakukan sebelumnya dengan *point addition* yang nantinya akan dilakukan. Nilai s *point addition* didapatkan dengan cara berikut.

3P = 2P + P = (6,3) + (5,1) = (x3,y3)

s = (y2-y1) / (x2-x1) = (1-3) / (5-6) = -2/-1 = 2

Sama seperti sebelumnya nilai s perlu disesuaikan dengan (mod 17). Nilai s yang telah didapatkan digunakan untuk mencari nilai (x3,y3) untuk 3P. Begitu seterusnya untuk mendapatkan nilai 4P, 5P, 6P, 7P, … . Proses *point doubling* dan *point addition* diperlukan untuk mendapatkan hasil akhir dari *point multiplication* nilai skalar k pada k*P dari ECC nantinya.




**GAMBAR 16: *ELLIPTIC CURVE DIFFIE-HELLMAN* (ECDH) KEY EXCHANGE**

Misalnya Angga akan mengirim pesan ke Budi menggunakan ECC. Maka dibutuhkan pertukaran kunci antara Angga dan Budi. Gambar 16 menunjukkan pertukaran kunci Angga dan Budi. Dapat dilihat bahwa meskipun penyadap mendapatkan persamaan

EC, koordinat titik G dari EC yang digunakan, n, dan public key (A, B). Namun, private key Angga dan Budi tidak didapatkan oleh penyadap. Hasil komputasi *private key* dan *public key* akan menghasilkan nilai P yang sebenarnya merupakan *point multiplication* dari k*P atau nilai skalar k dikalikan dengan nilai asli P yang digunakan sebagai kunci enkripsi pesan nantinya Nilai akhir P ini dimiliki oleh Angga dan Budi sehingga keduanya dapat bertukar pesan dengan kunci yang tidak dimiliki oleh penyadap. Pertukaran kunci ini disebut dengan *Diffie-Hellman key exchange*.

Tanda panah pada satu arah gambar 16 menunjukkan pertukaran public key dari Angga ke Budi dan Budi ke Angga. Sedangkan tanda panah dua arah menunjukkan nilai variabel yang disetujui kedua belah pihak. Karena data yang dipertukarkan ini diasumsikan melalui saluran yang telah disadap penyadap maka data EC, G, n, A, dan B juga diasumsikan telah diketahui oleh penyadap. Untuk nilai data yang ditulis tebal (α, β, dan P) tidak dikirimkan maka diasumsikan nilai ini tidak diketahui penyadap.

Nilai A dan B yang merupakan perkalian dari kunci privat (α atau β) dengan base point (G) walaupun diketahui penyadap, tetapi nilai ini adalah nilai akhir hasil perkalian. Kombinasi dari perkalian kunci privat dan *base point* sangat besar jumlahnya sehingga nilai kunci privat (α atau β) tidak pernah diketahui oleh penyadap.

Enkripsi ECC membutuhkan resource kalkulasi yang lebih kecil dibadingkan dengan pengamanan enkripsi dengan menggunakan enkripsi standar saat ini (contohnya RSA). Dengan menggunakan mesin komputasi yang sama dan kebutuhan *resources* komputasi lebih kecil, kalkulasi ECC tentunya akan lebih cepat dibandingkan dengan RSA, tetapi dengan hasil *cipher* yang sama amannya [2]. Perbandingan keamanan ini ditunjukkan pada tabel 2 [7]. Hal ini sangat bermanfaat bagi devais yang memiliki *resource* terbatas seperti contohnya tablet dan smartphone.

**TABEL 2: NIST KEY SIZE [7]**

| Symmetric key size (bits) | RSA Diffie-Hellman key size (bits) | Elliptic Curve key size (bits) |
|---:|---:|---:|
| 80 | 1024 | 160 |
| 112 | 2048 | 224 |
| 128 | 3072 | 256 |
| 192 | 7680 | 384 |
| 256 | 15360 | 521 |

## 3. SIGNCRYPTION WITH ELLIPTICAL CURVE CRYPTOGRAPHY (ECC)

Seperti yang telah dijabarkan secara eksplisit pada judul, konsep signcryption yang telah dijelaskan sebelumnya digunakan sebagai dasar blok signcryption. Perbedaannya pada generate public key dan private key yang menggunakan ECC. Tabel 3 adalah parameter signcryption dengan ECC [8].

**TABEL 3: PARAMETER PUBLIK YANG DIPERTUKARKAN**

| Variabel | Keterangan |
|---|---|
| C | Persamaan EC. |
| q | Bilangan prima dengan nilai besar. |
| G | Titik pada EC yang dipilih secara acak. |
| r, s | Hasil hash private key A dan public key B atau sebaliknya. Hasil hash dibagi menjadi 2 key. |

**TABEL 4: PARAMETER RAHASIA MILIK ANGGA**

| | |
|---|---|
| vb = β | private key Budi |
| P | shared secret key Angga dan Budi |

**TABEL 5: PARAMETER RAHASIA MILIK BUDI**

| | |
|---|---|
| va = α | private key Angga |
| P | shared secret key Angga dan Budi |

Signcryption akan membuat pengirim mentransmisikan 3 bentuk dari pesan yaitu c (cipher text), r (verifikasi dan hashing), dan s (hashing). Formula kalkulasi r dan s digunakan saat ini harus memenuhi salah satu dari digital signature standard (DSS) tabel 7 [8].

**TABEL 6: LANGKAH IMPLEMENTASI SIGNCRYPTION PADA EC**

| 1 | 2 | 3 |
|---|---|---|
| Signcryption pesan m oleh Angga sebagai pengirim | | Unsigncryption c,r,s oleh Budi sebagai penerima |
| v ∈ R(1, …, q-1)<br>(k1, k2) = hash(v*Pb)<br>c = Encrypt(k1(m))<br>r = KeyedHash(k2(m))<br><br>SECDSS1<br>s = (v/(r + va)) mod q<br><br>SECDSS2<br>s = (v/(1 + va*r)) mod q | (c, r, s)<br>==> | u = s*vb mod q<br><br>SECDSS1<br>(k1, k2) = hash(u*Pa + u*r*G)<br><br>SECDSS2<br>(k1, k2) = hash(u*G + u*r*Pa)<br><br>m = Decrypt(k1(c))<br><br>Pesan valid:<br>KeyedHash(k2(m)) = r |

**TABEL 7: VARIASI ELLIPTIC CURVE DSS**

| Signature (r, s) | Verifikasi signature | Panjang signature |
|---|---|---|
| ECDSS<br>r = v*G (mod q)<br>s = ((hash(m) + va * r)/v) mod q | s' = (1/s) mod q<br>K = s'(hash(m) * G + r * Pa)<br><br>verifikasi:<br>K mod q = r | 2 * lql |
| SCDSS1<br>r = hash(v*G, m)<br>s = (v/(r + va)) mod q | K = s(Pa + r*G)<br><br>verifikasi:<br>hash(K, m) = r | lhash(.)l + lql |
| SCDSS2<br>r = hash(v*G, m)<br>s = (v/(1 + va*r)) mod q | K = s(G + r*Pa)<br><br>verifikasi:<br>hash(K, m) = r | lhash(.)l + lql |

Perbandingan *computational cost* antara sign then encrypt menggunakan ECC dibandingkan dengan signcryption ECC berdasarkan [8] computational cost-nya berkurang sebesar 58%. Sedangkan communication cost berkurang sebesar 40%.

# DAFTAR PUSTAKA

[1]. Bakardjieva, Teodora. *Digital Signature*. http://vfu.bg/en/e-Learning/E-Business--DigSig.ppt. Diakses 29 Mei 2015.

[2]. Sullivan, Nick. *A (Relatively Easy To Understand) Primer on Elliptic Curve Cryptography*. https://blog.cloudflare.com/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/. Diakses 15 Juni 2015

[3]. Saini and Vaisla. *Image Signcryption using ECC*. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=7065597&abstractAccess=no&userType=inst. Diakses 29 Mei 2015.

[4]. Iqbal, Waseem. *An Efficient Elliptic Curve Based Signcryption Scheme for Firewall*. http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6725326. Diakses 28 Mei 2015.

[5]. Bedi, Harkeerat. *Elliptic Curve Cryptography*. web2.utc.edu/~djy471/CNS.../ECC%20-%20Harkeerat%20Bedi.ppt. Diakses 17 Juni 2015.

[6]. Paar, Christof and Pelzl. 2010. *Understanding Cryptography*. Heidelberg. Penerbit Springer.

[7]. National Security Agency. 2009. *The Case for Elliptic Curve Cryptography*. https://www.nsa.gov/business/programs/elliptic_curve.shtml. Diakses 17 Juni 2015.

[8]. Zhang, Yuliang and Imai. 1996. *Efficient Signcryption Schemes On Elliptic Curves*. http://webpages.uncc.edu/yzheng/publications/files/zheng-imai-ifip98-fnl.pdf. Diakses 19 Juni 2015.