Skema dan Arsitektur *Digital Signature* untuk Aplikasi *Mobile*

Adhitya Randy – NIM : 13503027

Laboratorium Ilmu dan Rekayasa Komputasi
Progam Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail: if13027@students.if.itb.ac.id

Abstrak

Perangkat dan aplikasi *mobile* sering digunakan untuk berbagai macam bidang kehidupan saat ini. Penggunaan perangkat dan aplikasi *mobile* untuk berbagai kegiatan yang penting mendorong munculnya suatu metode untuk menjaga keamanan pengguna dan seluruh pihak yang terlibat. Salah satu metode pengamanan yang sering digunakan untuk permasalahan di atas adalah penggunaan *digital signature*.

Perangkat *mobile* biasanya mempunyai keterbatasan seperti terbatasnya energi, memori, dan kemampuan komputasi yang dapat digunakan. Karena itulah perancangan skema dan arsitektur *digital signature* untuk aplikasi *mobile* sangat penting. Dalam studi ini akan dibahas skema dan arsitektur *digital signature* untuk aplikasi *mobile*. Terdapat dua pendekatan perancangan skema dan arsitektur *digital signature* tersebut yaitu berbasiskan *server* dan berbasiskan perangkat *mobile*. Skema dan arsitektur yang digunakan juga harus dirancang sebaik mungkin agar hemat energi, memori, dan komputasi. Selain itu, dalam studi ini juga akan dibahas penerapan *digital signature* untuk proses autentikasi pada aplikasi *mobile*.

***Kata kunci**: Digital signature, skema, arsitektur, aplikasi mobile*

1 Pendahuluan

Perangkat dan aplikasi *mobile* sering tidak dapat dipisahkan lagi dari dalam kehidupan manusia saat ini. Penggunaan perangkat dan aplikasi *mobile* saat ini sangat luas di dalam berbagai macam bidang kehidupan. Bahkan banyak kegiatan penting dilakukan hanya dengan menggunakan perangkat dan aplikasi *mobile*, misalnya saja untuk kegiatan bisnis atau perbankan. Karakteristik perangkat *mobile* yang tergabung dalam jaringan dengan menggunakan gelombang radio mengakibatkan setiap data yang dikirimkannya dapat diambil oleh pihak yang tidak berkepentingan. Karakteristik ini mendorong munculnya suatu metode pengamanan setiap pihak yang terlibat dalam penggunaan perangkat dan aplikasi *mobile*. Selain itu, dalam penerapan kegiatan penting pada perangkat dan aplikasi *mobile*, seperti untuk kegiatan bisnis dan perbankan, diperlukan suatu mekanisme autentikasi setiap pihak yang terlibat dengan cara yang mudah, cepat, efisien, dan aman. Pentingnya pengamanan pada aplikasi *mobile* ini ditujukan agar para pengguna dapat percaya dalam penggunaan aplikasi ini. Terlebih lagi pada penggunaan aplikasi seperti perbankan, saham, jual beli, atau aplikasi bernilai tinggi lainnya, diperlukan suatu tingkat keamanan yang dapat dipercaya.

Kriteria keamanan dan autentikasi yang dibutuhkan dalam mendukung dan menjaga keutuhan kegiatan yang berlangsung dengan menggunakan perangkat dan aplikasi *mobile* dapat dipenuhi dengan menggunakan *digital signature*. *Digital signature* atau tanda tangan digital dapat digunakan untuk memenuhi dua kriteria di atas karena *digital signature* atau tanda tangan dijital mempunyai karakteristik dapat menjamin integritas data, pembuktian asal pesan (keabsahan pengirim), serta nirpenyangkalan.

Proses pemberian tanda tangan dijital saat ini biasanya menggunakan algoritma kunci publik atau menggunakan fungsi *hash*. Baik algoritma kunci publik maupun fungsi *hash* biasanya memerlukan memori dan kemampuan komputasi mesin yang cukup tinggi. Proses komputasi yang tinggi dan penggunaan memori yang cukup besar akan menghabiskan energi yang besar. Di lain pihak, perangkat *mobile* biasanya mempunyai keterbatasan seperti terbatasnya energi, memori, dan kemampuan komputasi yang dapat digunakan. Karakteristik perangkat mobile dan tanda tangan dijital yang saling berlawanan ini menyebabkan tanda tangan dijital sulit diaplikasikan pada perangkat *mobile*. Karena itulah diperlukan suatu skema dan arsitektur tanda tangan dijital untuk aplikasi *mobile*. Tujuan dari skema dan arsitektur ini adalah agar aplikasi *mobile* dapat menerapkan tanda tangan dijital untuk berbagai macam keperluan walaupun perangkat *mobile* yang digunakan mempunyai keterbatasan energi, memori, dan kemampuan komputasi yang dapat digunakan.

2 Dasar Teori

2.1 Konsep Tanda Tangan Dijital

Tanda tangan telah digunakan sejak lama untuk membuktikan autentikasi, kepemilikan, dan keabsahan suatu dokumen kertas. Karakteristik yang menarik dari suatu tanda tangan adalah sebagai berikut [SCH96]:

1. Tanda tangan adalah bukti yang autentik. Tanda tangan meyakinkan penerima dokumen bahwa tanda tangan tersebut dibubuhkan pemiliknya secara terencana.
2. Tanda tangan tidak dapat ditiru. Tanda tangan membuktikan bahwa sang penandatangan, bukan orang lain yang menandatangani dokumen secara terencana.
3. Tanda tangan tidak dapat dipindah atau digunakan ulang. Tanda tangan merupakan bagian dari dokumen sehingga tidak ada satu orang pun yang dapat memindahkan sebuah tanda tangan dari satu dokumen ke dokumen lainnya yang berbeda.
4. Dokumen yang telah ditandatangani tidak dapat diubah. Setelah ditandatangani, suatu dokumen tidak dapat diubah kembali.
5. Tanda tangan tidak dapat disangkal (*repudiation*). Tanda tangan dan dokumen adalah benda fisik. Sang penandatangan tidak dapat menyangkal bahwa ia tidak menandatangani dokumen tersebut.

Walaupun sebuah tanda tangan mempunyai karakteristik-karakteristik seperti yang telah disebutkan di atas, pada kenyataannya tidak semua pernyataan di atas benar. Tanda tangan dapat ditiru, tanda tangan dapat dipindahkan dari selembar kertas suatu dokumen dan dipindahkan ke lembar kertas lainnya, dan suatu dokumen tetap dapat diubah setelah ditandatangani.

Walaupun mempunyai kelemahan, tanda tangan ini juga dapat diterapkan pada komputer dan data dijital. Tetapi penerapan tanda tangan pada komputer ini mempunyai beberapa masalah. Permasalahan pertama adalah arsip-arsip komputer mudah untuk diperbanyak. Walaupun tanda tangan seseorang sulit untuk ditiru (misalnya sebuah citra grafis dari tanda tangan), sangatlah untuk memperbanyak tanda tangan yang valid tersebut dari satu dokumen ke dokumen lainnya. Hal ini menyebabkan hanya keberadaan dari sebuah tanda tangan tidak berarti apapun. Permasalahan kedua adalah arsip-arsip komputer sangat mudah dimodifikasi setelah ditandatangani, tanpa meninggalkan jejak modifikasi.

Walaupun begitu fungsi tanda tangan pada dokumen kertas juga dapat diterapkan untuk autentikasi pada data dijital seperti pesan yang dikirim melalui saluran komunikasi dan dokumen elektronis yang terseimpan di dalam memori komputer. Tanda tangan pada data dijital ini dinamakan tanda tangan dijital (*digital signature*). Yang dimaksud dengan tanda tangan dijital bukanlah tanda tangan tertulis yang digitalisasi, misalnya dengan alat *scanner*, melainkan suatu nilai kriptografis yang bergantung pada pesan dan pengirim pesan. Hal ini kontras dengan tanda tangan pada dokumen kertas yang bergantung hanya pada pengirim dan selalu sama untuk semua dokumen. Dengan tanda tangan dijital, maka integritas data dapat dijamin. Selain itu, tanda tangan dijital juga dapat digunakan untuk membuktikan asal pesan (keabsahan pengirim), dan nirpenyangkalan.

Menandatangani suatu pesan atau data dijital dapat dilakukan dengan kedua cara berikut:

1. Tanda tangan dijital dengan enkripsi pesan.
   Mengenkripsi pesan dengan sendirinya juga menyediakan ukuran autentikasi. Pesan yang terenkripsi sudah menyatakan bahwa pesan tersebut telah ditandatangani.
2. Tanda tangan dijital dengan fungsi *hash*.
   Tanda tangan dijital dibangkitkan dari *hash* terhadap pesan. Nilai *hash* merupakan kode ringkas dari pesan. Tanda tangan dijital berlaku seperti tanda tangan pada dokumen kertas. Tanda tangan ini kemudian ditambahkan (*append*) pada pesan.

2.2 Penandatanganan dengan Mengenkripsi Pesan

2.2.1 Menandatangani Pesan dengan Algoritma Kriptografi Kunci Simetri

Pesan yang dienkripsi dengan algoritma kriptografi kunci simetri sebenarnya sudah memberikan solusi untuk autentikasi pengirim dan keaslian pesan, karena kunci simetri hanya diketahui oleh pengirim dan penerima saja. Misalnya jika Bob menerima pesan dari Alice, maka ia percaya pesan itu berasal dari Alice. Hal ini disebabkahn karena selain Bob hanya Alice yang mengetahui kunci rahasia K sehingga ketika pesan didekripsi dengan kunci K hasilnya adalah pesan bermakna. Pihak ketiga yang tidak mengetahui K tidak dapat menghasilkan cipherteks lain yang dapat didekripsi dengan K. Selanjutnya, Bob yakin isi pesan tidak diubah selama transmisi sebab pihak ketiga yang tidak mengetahui K tidak tahu cara mengubah data pada cipherteks untuk menghasilkan perubahan yang diinginkan pada plainteks.

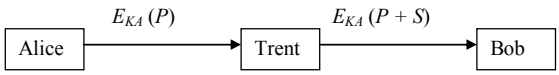
Namun, algoritma kriptografi kunci simetri tidak dapat menyediakan suatu mekanisme untuk mengatasi masalah penyangkalan, yaitu jika salah satu dari dua pihak, Alice dan Bob, membantah isi pesan atau telah mengirim pesan. Karena hanya

Alice dan Bob yang mengetahui kunci rahasia, maka Alice dapat menyangkal dengan mengatakan bahwa Bob telah mengirim pesan tersebut, demikian pula dengan Bob dapat menyangkal dengan mengatakan bahwa pean tersebut dibuat oleh Alice. Pada kasus ini tidak dapat dilakukan nirpenyangkalan untuk membuktikan bahwa salah satu pihak sudah berbohong.

Agar dapat melakukan nirpenyangkalan dengan algoritma kriptografi kunci simetri, maka diperlukan pihak ketiga yang dipercaya oleh pihak pengirim dan penerima. Pihak ketiga ini disebut penengah (arbitrase). Sebagai contoh, misalnya Trent adalah otoritas arbitrase yang dapat dipercaya oleh Alice dan Bob. Trent memberikan kunci rahasia K_A kepada Alice dan kunci rahasia K_B kepada Bob. Hanya Alice dan Trent yang mengetahui K_A, begitu juga hanya Bob dan Trent yang mengetahui K_B. Kunci-kunci ini telah ditetapkan sebelum protokol tanda tangan dimulai dan dapat digunakan berulang kali untuk tanda tangan berulang kali. Jika Alice ingin mengirimkan pesan P kepada Bob, maka langkah-langkah yang harus dilakukan adalah sebagai berikut:

1. Alice mengenkripsikan pesannya P kepada Bob dengan menggunakan K_A dan mengirimkannya kepada Trent.
2. Trent mendekripsikan pesan P tersebut dengan menggunakan K_A.
3. Trent mengambil pesan yang telah didekripsikan dan membuat pernyataan S bahwa ia telah menerima pesan dari Alice. Pernyataan S ini kemudian ditambahkan pada pesan yang telah didekripsikan ini. Trent selanjutnya mengenkripsikan bundel pesan P dan S dengan kunci K_B.
4. Trent kemudian mengirimkan bundel yang telah terenkripsi kepada Bob.
5. Bob mendekripsikan bundel pesan tersebut dengan menggunakan K_B. Bob kemudian dapat membaca pesan P dari Alice dan pernyataan S dari Trent yang menyatakan Alice mengirimkan pesan P tersebut.

Gambar 1 menunjukkan skema penandatanganan dengan menggunakan kunci simetri.



Gambar 1. Penandatanganan Pesan Algoritma Kriptografi Kunci Simetri dan Dengan Bantuan Arbitrase

Trent dapat mengetahui pesan yang dikirim berasal dari Alice dan bukan dari pihak ketiga yang tidak dikenal melalui proses enkripsi pesan tersebut. Karena hanya Trent dan Alice yang mempunyai kunci K_A, maka hanya Alice yang dapat mengenkripsi pesan dengan menggunakannya.

Karakteristik dari proses penandatanganan ini adalah sebagai berikut:

1. Tanda tangan tersebut autentik. Trent adalah arbitrase yang terpercaya dan Trent mengetahui bahwa pesan tersebut datang dari Alice. Pernyataan Trent tersebut menjadi bukti bagi Bob.
2. Tanda tangan tersebut tidak dapat ditiru. Hanya Alice (dan Trent, tetapi semua pihak mempercayainya) mengetahui K_A, jadi hanya Alice yang dapat mengirimkan pesan yang dienkripsikan dengan K_A kepada Trent. Jika seseorang mencoba untuk menyamar menjadi Alice, Trent dapat mengetahuinya pada langkah 2 pada proses penandatanganan dan tidak akan memberikan pernyataan autentikasi pesan dari Alice tersebut.
3. Tanda tangan tersebut tidak dapat digunakan ulang. Jika Bob mencoba untuk mengambil pernyataan Trent dan menambahkannya pada pesan lainnya, Alice dapat menyangkalnya. Seorang arbitrasi dapat meminta Bob untuk memproduksikan baik plainteks maupun cipherteks dari Alice. Arbitrase ini kemudian akan mencoba mengenkripsikan plainteks dengan K_A dan menghasilkan cipherteks yang tidak sama dengan cipherteks yang diberikan oleh Bob. Tentu saja Bob tidak dapat membuat pesan yang terenkripsi dengan tepat karena dia tidak mengetahui K_A.
4. Dokumen yang telah ditandatangani tidak dapat diubah kembali. Jika Bob mencoba untuk mengubah dokumen yang telah diterimanya, Trent dapat membuktikannya dengan cara seperti yang telah dijelaskan di atas.
5. Tanda tangan tersebut tidak dapat disangkal. Walaupun Alice kemudian menyangkal bahwa ia telah mengirimkan pesan, pernyataan dari Trent menyatakan yang sebaliknya. Karena Trent adalah arbitrase yang dipercayai oleh semua orang, maka pernyataan yang dikeluarkannya adalah benar.

Jika Bob ingin menunjukkan pada Carol sebuah dokumen yang telah ditandatangani Alice, Bob tidak dapat menunjukkan kuncinya pada Carol. Bob harus tetap melakukannya melalui Trent yaitu dengan langkah-langkah sebagai berikut:

1. Bob mengambil pesan dan pernyataan Trent bahwa pesan tersebut berasal dari Alice, kemudian mengenkripsikannya dengan K_B, dan mengirimkannya kembali kepada Trent.
2. Trent mendekripsikan bundel pesan tersebut dengan menggunakan KB.
3. Trent mengecek pada basisdata dan mengkonfirmasi bahwa pesan tersebut berasal dari Alice.

4. Trent mengenkripsi kembali bundel pesan tersebut dengan K_C (kunci yang hanya diketahui Trent dan Carol) dan mengirimkannya kepada Carol.
5. Carol mendekripsikan bundel tersebut dengan menggunakan K_C. Carol kemudian dapat membaca pesan dan pernyataan Trent bahwa Alice mengirimkan pesan tersebut.

Cara penandatanganan ini dapat berjalan tetapi sangat menghabiskan waktu bagi Trent. Trent harus meluangkan waktunya untuk mendekripsikan dan mengenkripsikan pesan-pesan, bertindak sebagai penengah antara setiap pasangan orang yang ingin mengirimkan pesan bertanda tangan kepada orang lainnya. Trent juga harus menyimpan basis data pesan-pesan (walaupun cara ini dapat dihindari dengan mengirimkan salinan cipherteks dari pengirim kepada penerima). Selain itu, Trent juga merupakan *bottleneck* pada sistem komunikasi walaupun ia merupakan progam atau perangkat lunak.

Selain itu sangatlah sulit membuat dan memelihara seseorang seperti Trent, seseorang yang dapat dipercaya setiap orang dalam jaringan. Trent tidak dapat membuat kesalahan sedikitpun. Jika Trent membuat satu buah kesalahan dalam jutaan penandatanganan, tidak akan ada seorang pun yang akan mempercayainya kembali. Selain itu Trent juga haruslah sangat aman. Basis data kunci dan program arbitrase ini haruslah sangat aman. Karena kesulitan-kesulitan inilah cara penandatanganan ini tidak dapat diterapkan di dunia nyata.

2.2.2 Menandatangani Pesan dengan Algoritma Kriptografi Kunci Publik

Jika algoritma kriptografi kunci publik digunakan utnuk menandatangani pesan, maka enkripsi pesan dengan kunci publik tidak dapat digunakan untuk autentikasi, karena setiap orang potensial mengetahui kunci publik. Tetapi, jika enkripsi pesan menggunakan kunci privat pengirimdan dekripsi pesan menggunakan kunci publik si pengirim, maka kerahasiaan pesan dan autentikasi dapat dicapai sekaligus.

Beberapa algoritma kunci-publik dapat digunakan untuk menandatangani pesan dengan cara mengenkripsinya, asalkan algoritma tersebut memenuhi sifat: $D_{SK}(E_{PK}(M)) = M$ dan $D_{PK}(E_{SK}(M)) = M$, dengan PK = kunci publik dan SK = kunci privat (*secret key*).

Misalkan M adalah pesan yang akan dikirim, Pesan M ditandatangani menjadi pesan terenkripsi S dengan menggunakan kunci privat (SK) pengirim.

$$S = E_{SK}(M)$$

Dalam hal ini E adalah fungsi enkripsi dari algoritma kriptografi kunci publik. Selanjutnya S dikirim melalui saluran komunikasi. Di tempat penerima pesan dibuktikan autentikasinya dengan menggunakan kunci publik (PK) pengirim.

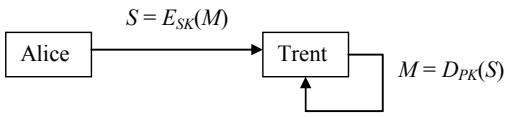
$$M = D_{PK}(S)$$

Dalam hal ini D adalah fungsi dekripsi dari algoritma kriptografi kunci publik. S dikatakan absah apabila pesan M yang dihasilkan merupakan pesan yang mempunyai makna.

Langkah-langkah yang harus dilakukan dalam proses pemberian tanda tangan dijital dengan menggunakan algoritma kriptografi kunci publik adalah sebagai berikut:
1. Alice mengenkripsikan dokumen dengan kunci privat miliknya (proses penandatanganan dokumen).
2. Alice mengirimkan dokumen yang telah ditandatangani kepada Bob.
3. Bob mendekripsikan dokumen yang dikirimkan dengan menggunakan kunci publik Alice (proses verifikasi tanda tangan).

Penggunaan algoritma kriptografi kunci publik untuk proses pemberian tanda tangan dijital tidak memerlukan lagi pihak penengah seperti pada penandatanganan dengan menggunakan algoritma kriptografi kunci simetri. Jika Bob tidak dapat melakukan langkah 3, maka tanda tangan tersebut tidak valid.

Gambar 2 menunjukkan skema penandatanganan dengan menggunakan kunci simetri



Gambar 2. Penandatanganan Pesan Algoritma Kriptografi Kunci Publik

Cara penandatanganan pesan dengan alogritma kunci publik ini memenuhi karakteristik-karakteristik tanda tangan yang diinginkan, yaitu sebagai berikut:
1. Tanda tangan tersebut autentik. Ketika Bob memverifikasi pesan dengan menggunakan kunci publik Alice, Bob dapat mengetahui bahwa Alice yang menandatangani dokumen tersebut.
2. Tanda tangan tersebut tidak dapat ditiru. Hal ini disebabkan karena hanya Alice yang mengetahui kunci privat yang digunakan untuk tanda tangan.
3. Tanda tangan tersebut tidak dapat digunakan ulang. Tanda tangan merupakan fungsi dari

dokumen dan tidak dapat dipindahkan ke dokumen lainnya.
4. Dokumen yang telah ditandatangani tidak dapat diubah kembali. Jika terdapat perubahan pada dokumen, tanda tangan tidak dapat diverifikasi dengan menggunakan kunci publik Alice yang bersesuaian.
5. Tanda tangan tersebut tidak dapat disangkal. Bob tidak memerlukan bantuan Alice untuk memverifikasi tanda tangan Alice.

2.3 Penandatangan dengan Fungsi Hash

Dalam implementasinya, algoritma kriptografi kunci publik biasanya tidak efisien dalam menandatangani dokumen yang sangat panjang. Selain itu penandatanganan pesan dengan cara mengenkripsinya selalu memberikan dua fungsi berbeda, yaitu kerahasiaan dan autentikasi. Sedangkan pada beberapa kasus, seringkali fungsi autentikasi saja yang diperlukan dan fungsi kerahasiaan pesan tidak. Dengan kata lain, pesan tidak perlu dienkripsikan, sebab yang dibutuhkan hanya autentikasi saja.

Hal-hal di atas memunculkan ide bahwa daripada menandatangani dokumen, maka pengguna cukup mendatangani hasil *hash* dari dokumen saja. Hanya sistem kriptografi kunci publik yang cocok dan alami untuk pemberian tanda tangan dijital dengan menggunakan fungsi *hash*. Hal ini disebabkan karena skema tanda tangan dijital berbasis sistem kriptografi kunci publik dapat menyelesaikan masalah *non-repudation*.

2.3.1 Fungsi *Hash* Satu Arah

Fungsi hash satu arah juga dikenal dengan banyak nama, misalnya fungsi kompresi, fungsi kontraksi, pesan singkat (*message digest*), sidik jari (*fingerprint*), *cryptographic checksum*, *message integrity check* (MIC), dan *manipulation detection code* (MDC). Fungsi hash merupakan pusat dari kriptografi modern karena fungsi *hash* ini merupakan blok pembangan untuk berbagai macam protokol.

Fungsi *hash* telah lama digunakan dalam bidang keilmuan komputer. Sebuah fungsi *hash* adalah sebuah fungsi (secara matematis) yang menerima input string dengan panjang bervariasi (*pre-image*) dan mengkonversinya kepada sebuah output string (*hash value*) yang mempunyai panjang tetap (biasanya lebih pendek). Sebuah fungsi *hash* sederhana dapat dibangun dengan cara menerima *pre-image* dan mengembalikan sebuah *byte* yang berasal dari operasi XOR terhadap seluruh *byte* dari *pre-image*.

Pada intinya, fungsi *hash* berguna untuk menghasilkan suatu nilai yang mengindikasikan sebuah kandidat *pre-image* memiliki kemungkinan sama dengan *pre-image* sebenarnya. Karena fungsi-fungsi *hash* pada dasarnya banyak yang bersifat memetakan dari banyak domain ke satu range, fungsi *hash* tidak dapat digunakan untuk menentukan bahwa kedua buah string adalah sama, tetapi dapat digunakan untuk mendapatkan tingkat kepastian akurasi.

Aplikasi fungsi *hash* misalnya untuk memverifikasi kesamaan salinan suatu arsip dengan arsip aslinya yang tersimpan di dalam sebuah basis data terpusat. Daripada mengirim salinan arsip tersebut secara keseluruhan ke komputer pusat (yang memerlukan waktu transmisi yang lama dan onkos yang mahal), lebih efisien mengirimkan *message-digest*-nya atau nilah *hash*-nya saja. Jika *message digest* salinan arsip sama dengan *message digest* arsip asli, berarti salinan arsip tersebut sama dengan arsip di dalam basis data.

Fungsi *hash* satu arah adalah sebuah fungsi *hash* yang berkerja dengan satu arah. Sangatlah mundah untuk menghitung sebuah nilai *hash* dari *pre-image*, tetapi sangatlah sulit untuk membuat sebuah *pre-image* yang mempunyai nilai *hash* yang sama dengan suatu nilai tertentu. Sebelumnya fungsi *hash* tidak disebutkan sebagai fungsi yang satu arah. Jika diberikan sebuah nilai *byte* tertentu, sangatlah mudah untuk membangun sebuah *byte-byte* sebuah string yang hasil dari operasi XOR adalah nilai tersebut. Sebuah fungsi *hash* satu arah tidak mempunyai karakteristik seperti itu. Fungsi *hash* satu arah yang baik juga mempunyai karakteristik *collision-free*, yang berarti sangatlah sulit untuk membangun dua buah *pre-image* berbeda yang mempunyai nilai *hash* yang sama.

Fungsi *hash* adalah algoritma publik, tidak ada kerahasiaan dalam prosesnya. Keamanan suatu fungsi *hash* didadapatkan melalui karakteristiknya yang satu arah. Hasil dari fungsi *hash* pun tidak mempunyai keterhubungan apapun dengan nilai masukan. Perubahan satu bit pada pre-image akan mengakibatkan perubahan kurang lebih setengah dari keseluruhan bit pada nilai *hash*. Jika diberikan sebuah nilai *hash*, sangat tidak mungkin secara komputasi untuk menemukan sebuah *pre-image* yang mempunyai nilai hash yang sama dengan nilai tersebut.

Sifat-sifat fungsi *hash H* satu arah adalah sebagai berikut:
1. Fungsi *H* dapat diterapkan pada blok data berukuran berapa saja.
2. *H* menghasilkan nilai (*h*) dengan panjang tetap (*fixed-length output*).

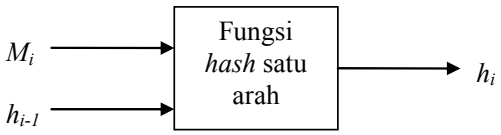
3. $H(x)$ mudah dihitung untuk setiap nilai yang diberikan.
4. Untuk setiap h yang diberikan, tidak mungkin menemukan x sedemikian sehingga $H(x) = h$. Itulah sebabnya fungsi H dikatakan fungsi hash satu arah (*one way hash function*).
5. Untuk setiap x yang diberikan, tidak mungkin mencari nilai $y \neq x$ sedemikian sehingga $H(x) = H(y)$.
6. Tidak mungkin (secara komputasi) mencari pasangan x dan y sedemikian sehingga $H(x) = H(y)$.

Keenam sifat di atas penting sebab sebuah fungsi *hash* seharusnya berlakuk seperti fungsi acak. Sebuah fungsi *hash* dianggap tidak aman jika secara komputasi dimungkinkan menemukan pesan yang bersesuaian dengan pesan ringkasnya dan terjadi kolisi (*collision*), yaitu terdapat beberapa pesan yang berbeda tetapi mempunyai pesan ringkas atau nilai *hash* yang sama.

Fungsi *hash* berkerja secara iteratif. Masukan fungsi *hash* adalah blok pesan (M) dan keluaran dari proses *hash* blok pesan sebelumnya.

$$h_i = H(M_i, h_{i-1})$$

Skema fungsi *hash* ditunjukkan pada Gambar 3.



Gambar 3. Skema Fungsi *Hash* Satu Arah

2.3.2 Proses Pemberian Tanda Tangan (*Signing*)

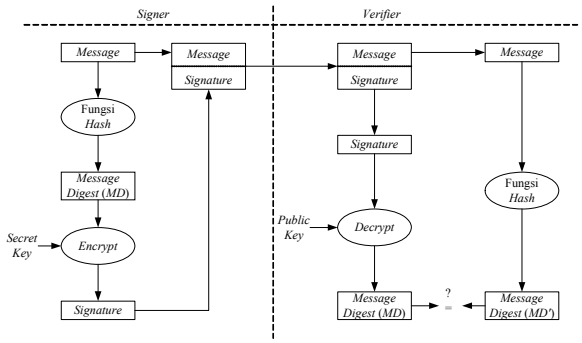
Langkah-langkah pemberian tanda tangan dijital adalah sebagai berikut:

1. Pengirim pesan menghitung *message digest* dari pesan. Pesan yang diubah terlebih dahulu menjadi *message digest MD* dengan menggunakan fungsi hash H satu arah.
$$MD = H(M)$$
2. *Message digest MD* dienkripsikan dengan algoritma kunci publik menggunakan kunci rahasia (*SK*) pengirim. Hasil enkripsi ini menjadi tanda tangan dijital S.
$$S = E_{SK}(MD)$$
3. Pesan M disambung (*append*) dengan tanda tangan dijital S.

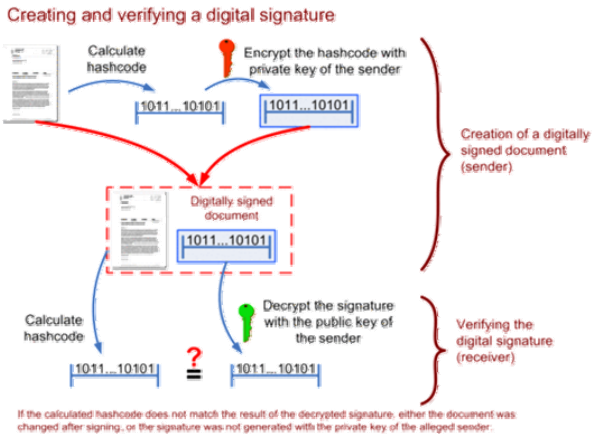
Selanjutnya, langkah-langkah untuk melakukan autentikasi oleh penerima adalah sebagai berikut:

1. Tanda tangan dijital S didekripsi dengan menggunakan kunci publik (*PK*) pengirim pesan, menghasilkan *message digest* semula, yaitu *MD*.
$$MD = D_{PK}(S)$$
2. Pesan M diubah menjadi *message digest MD*' menggunakan fungsi *hash* satu arah yang sama dengan fungsi *hash* yang digunakan oleh pengirim.
3. Jika *MD*' = *MD*, berarti pesan yang diterima otentik dan berasal dari pengirim yang benar.

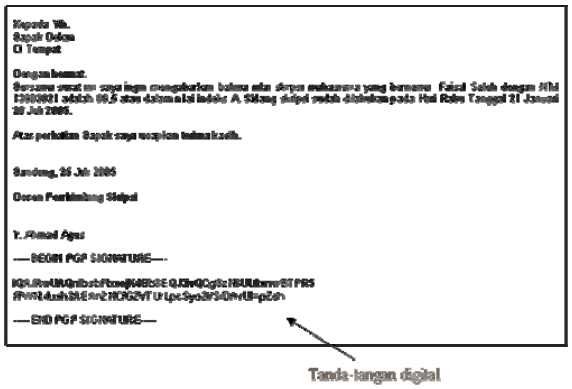
Skema tanda tangan dijital yang menggunakan fungsi *hash* ditunjukkan pada Gambar 4. Gambar 5 memvisualisasikan penandatanganan dan pemverifikasian tanda tangan. Sedangkan contoh dokumen yang telah diberikan tanda tangan dijital dapat dilihat pada Gambar 6.



Gambar 4. Otentikasi dengan Tanda Tangan Dijital Menggunakan Fungsi *Hash* Satu Arah



Gambar 5. Visualisasi Pemberian Tanda Tangan Dijital dan Verifikasinya



Gambar 6. Contoh Dokumen yang Dibubuhi Tanda Tangan Dijital

Autentikasi pesan dapat dijelaskan sebagai berikut:

1. Apabila pesan *M* yang diterima telah berubah, *MD'* yang dihasilkan dari fungsi *hash* berbeda degan *MD* semula. Ini berarti pesan tidak asli lagi.
2. Apabila pesan *M* tidak berasal dari orang sebenarnya, maka *message digest MD* yang dihasilkan akan berbeda dengan *message digest MD'* yang dihasilkan proses verifikasi. Hal ini disebabkan karena kunci publik yang digunakan oleh penerima pesan tidak berkorespondensi dengan kunci privat pengirim.
3. Bila *MD = MD'* berarti pesan yang diterima adalah pesan yang asli (*message authentication*) dan orang yang mengirim adalah orang yang sebenarnya (*user authentication*).

Pengirim pesan tidak dapat menyangkal pesan yang ia kirim sebab tanda tangan dijital dapat digunakan untuk melakukan nirpenyangkalan. Andaikan pengirim berbohong telah mengirim pesan, sangkalan dari pengirim pesan dapat dibantah dengan cara sebagai berikut: jika ia tidak mengirim pesan, berarti ia tidak mengenkripsi *message digest* dari pesan dengan kunci privatnya. Faktanya, kunci publik yang berkoresponden dengan kunci privat pengirim menghasilkan *MD = MD'* yang berarti *message digest* memang benar dienkripsi oleh pengirim, sebab hanya pengirim yang mengetahui kunci privatnya sendiri.

3 Protokol Kriptografi

Kegunaan utama dari kriptografi adalah untuk menyelesaikan permasalahan yang berkaitan dengan keamanan, autentikasi, integritas, dan pihak-pihak yang tidak berkepentingan. Sebuah protokol adalah rangakaian langkah, berkaitan dengan dua atau lebih pihak, yang dirancang untuk menyelesaikan suatu persoalan. Rangkaian langkah pada sebuah protokol berarti protokol mempunyai urutan tertentu dari awal hingga akhir. Setiap langkah harus dieksekusi pada saatnya dan tidak ada langkah yang dapat dilakukuan sebelum langkah sebelumnya selesai dilakukan. Sebuah protokol juga dilakukan dengan melibatkan dua atau lebih pihak sehingga setidaknya diperlukan dua orang untuk menyelesaikan protokol. Selain itu protokol dirancang untuk menyelesaikan sebuah persoalan berarti protokol harus mencapai sesuatu yang diinginkan.

Protokol juga mempunyai karakteristik lain sebagai berikut:

1. Setiap orang yang terlibat di dalam protokol harus mengetahui protokolnya dan setiap langkah harus diikuti.
2. Setiap orang yang terlibat dalam protokol harus setuju untuk mengikutinya.
3. Protokol harus tidak ambigu, setiap langkah harus terdefinisi dengan baik dan tidak boleh ada celah yang dapat mengakibatkan kesalahpahaman.
4. Protokol harus lengkap, harus ada aksi-aksi tertentu yang harus dilakukan dalam setiap situasi yang mungkin.

Protokol kriptografi adalah protokol yang menggunakan kriptografi. Pihak-pihak yang terlibah dapat saling kenal dan saling mempercayai atau dapat tidak mengenal dan tidak saling percaya satu sama lain. Sebuah protokol kriptografi menggunakan beberapa algoritma kriptografi, tetapi tujuan utama dari protokol kriptografi biasanya lebih dari hanya keamanan saja. Pihak-pihak yang berpartisipasi dalam sebuah protokol mungkin ingin untuk membagi bagian dari rahasia untk menghitung sebuah nilai, membangkitkan rangkaian bilangan acak, meyakinkan pihak lainnya, atau mendandatangani sebuah kontrak. Inti dari penggunaan kriptografi dalam sebuah protokol adalah untuk mencegah atau mendeteksi penyadapan dan penipuan.

4 Protokol Tanda Tangan Dijital

Protokol tanda tangan dijital merupakan bagian dari protokol kriptografi karena dalam proses tanda tangan dijital digunakan beberapa algoritma kriptografi. Tujuan utama dari sebuah tanda tangan dijital telah dijelaskan pada bagian 2. Pada bagian 2 pun telah dijelaskan protokol tanda tangan dijital dengan menggunakan algoritma kriptografi kunci simetri, algoritma kriptografi kunci publik, dan fungsi *hash*.

Protokol tanda tangan dijital sangat penting untuk menjamin proses tanda tangan dan proses verifikasi dapat berjalan dengan baik serta dapat dicapai tujuan tanda tangan dijital itu sendiri. Sebuah protokol akan berkaitan dengan skema dan arsitektur penggunaan protokolnya. Begitu pula protokol tanda tangan dijital juga berkaitan dengan skema dan arsitektur tanda tangan dijital.

Tanda tangan dijital telah banyak digunakan dalam aplikasi *mobile* saat ini. Karena banyaknya keterbatasan pada perangkat *mobile* (seperti keterbatasan energi, memori, dan kemampuan komputasi) maka skema dan arsitektur tanda tangan dijital untuk aplikasi *mobile* harus dirancang sedemikian rupa dengan memperhatikan keterbatasan dari perangkat *mobile*. Walaupun harus memperhatikan keterbatasan perangkat *mobile*, skema dan arsitektur tanda tangan dijital untuk aplikasi *mobile* tetap harus dapat memenuhi tujuan tanda tangan dijital itu sendiri, yaitu dapat menjaga kerahasiaan, integritas data, otentikasi, dan nirpenyangkalan.

5 Pendekatan Perancangan Skema dan Arsitektur Tanda Tangan Dijital pada Aplikasi Mobile

Terdapat dua pendekatan perancangan skema dan arsitektur *digital signature* tersebut yaitu berbasiskan *server* dan berbasiskan *client* atau perangkat *mobile*. Pemberian tanda tangan pada pendekatan berbasiskan *server* dilakukan pada lingkungan *server* terpusat yang terdapat pada penyedia layanan aplikasi, misalnya penyedia layanan jaringan telepon seluler. Sedangkan tanda tangan dijital berbasiskan *client* dilakukan dengan cara pembuatan tanda tangan pada perangkat *mobile* sang pemberi tanda tangan, misalnya dengan memanfaatkan *smart card* [ROS04].

5.1 Pendekatan Berbasiskan Server

Tanda tangan berbasiskan *server* ini berarti tanda tangan dibangkitkan oleh penyedia layanan bagi pengguna tertentu [FRI03]. Pemberian tanda tangan dijital berbasiskan *server* dapat dibagi ke dalam tiga kelompok bergantung pada relasi kepercayaan antara *client* dan *server*. Secara spesifik, *server* yang digunakan dapat terbagi menjadi dapat dipercaya, tidak dapat dipercaya, atau dapat diverifikasi [GOY04].

Pada *server* kategori pertama, proses tanda tangan dimulai dengan langkah pengirim mengirimkan pesan autentik kepada *server*. Pesan autentik ini dapat berupa hasil *hash* dari pesan jika proses tanda tangan yang digunakan adalah dengan menggunakan fungsi *hash*. Untuk kategori server ini sebaiknya digunakan tanda tangan dengan menggunakan fungsi *hash* dan fungsi *hash* yang digunakan adalah MAC. MAC adalah fungsi *hash* satu arah yang menggunakan kunci rahasia (*secret key*) dalam pembangkitan nilai *hash*. Dengan kata lain nilai *hash* yang dihasilkan adalah fungsi dari pesan dan kunci. Kemudian sebuah *server proxy* mewakilkan pengirim untuk membuat tanda tangan dijital dengan menggunakan algoritma kriptografi kunci publik untuk pesan tersebut.

Pengirim tidak perlu melakukan operasi algoritma kriptografi kunci publik apapun. Pengirim hanya perlu menghitung nilai *hash* dari pesannya. Kekurangan dari kategori ini adalah proses pemberian tanda tangan dijital hanya dapat dilakukan jika pengirim pesan sangat mempercayai *server* yang digunakannya. Selain itu, *server* yang digunakan oleh pengirim sebenarnya dapat saja memalsukan tanda tangan dijital yang digunakan dan tindak kecurangan *server* tidak dapat dibuktikan oleh pengirim pesan.

Pada sisi lainnya terdapat kategori *server* yang sama sekali tidak dapat dipercaya oleh penggunanya. Kategori ini masih dapat digunakan untuk melakukan komputasi yang tidak terlalu sensitif terahadap masalah keamanan dari penggunanya. Sampai saat ini belum ada skema dan arsitektur yang aman dan dapat diimplementasikan untuk kategori ini.

Penandatanganan dengan bantuan *server* untuk kategori terakhir adalah dengan menggunakan *server* yang dapat diverivikasi atau *verifiable server*. Sebuah *verifiable server* adalah sebuah *server* yang tindak kecurangannya dapat dibuktikan terhadap sebuah arbitrasi. Pendekatan ini pada dasarnya berada di antara dua kategori di atas karena pada kasus ini *server* dapat melakukan kecurangan, tetapi pengguna dapat memiliki kemampuan untuk membuktikannya kepada pihak lainnya (misalnya pihak arbitrasi).

5.2 Pendekatan Berbasiskan Client

Selain dapat dibuat pada *server*, tanda tangan dijital pun dapat dibuat pada perangkat *mobile* dengan menggunakan perangkat pembuatan tanda tangan yang aman. Dengan menggunakan sebuah *smart card* untuk tanda tangan, yang disertifikasi oleh penyedia sertifikasi, dimasukkan ke dalam perangkat *mobile* yang biasanya sudah memiliki kartu SIM. Dengan cara ini maka proses pemberian tanda tangan dijital dapat dilaukan pada perangkat *mobile* oleh pengirim pesan [ROS04].

Proses pemberian tanda tangan ini dapat dilakukan dengan dua cara. Cara pertama adalah demgan mengganti kartu SIM yang digunakan dengan kartu tanda tangan. Cara pertama ini sangatlah tidak nyaman bagi penggunanya karena perangkat *mobile* harus dimatikan terlebih dahulu untuk penggantian kartu untuk pembuatan tanda tangan dan kemudian untuk menggunakan fungsionalitas perangkat *mobile* yang lainnya. Cara lainnya adalah dengan menggunakan tambahan pembaca kartu chip pada perangkat *mobile*. Pada cara kedua ini diperlukan perangkat *mobile* khusus yang mempunyai lebih dari satu slot untuk *smart card*.

Selain itu terdapat solusi lainnya yaitu sebuah *smart card* mempunyai fungsi dari SIM dan juga pembangkitan tanda tangan dijital yang aman. Cara ini dapat dicapai dengan cara menyediakan ruang sisa pada memori kartu SIM untuk tempat instalasi komponen pembangkitan tanda tangan yang dapat dilakukan kemudian. Selain itu juga terdapat cara lainnya yaitu dengan memasarkan kartu SIM dengan fungsionalitas tanda tangan dijital yang telah terinstal yang harus diinisialisasi dan diaktifkan oleh penggunanya.

6 Skema dan Arsitektur Tanda Tangan Dijital pada Aplikasi *Mobile*

6.1 Skema Tanda Tangan Dijital berbasiskan Server untuk One Time Signature

Server asisted one time signature (SAOTS) merupakan skema dan arsitektur pemberian tanda tangan dijital berbasiskan *server* untuk aplikasi *mobile*. Walaupun sangat mudah untuk diimplementasikan, skema SAOTS pada awalnya memerlukan tempat penyimpanan yang besar pada *server* dan penggunaan memori yang besar pada perangkat *mobile client*. Skema dan arsitektur SAOTS yang baru dapat mengurangi sekitar 80 kali penggunaan tempat penyimpanan pada *server*. Selain itu, penggunaan memori pada perangkat *mobile client* juga dapat dikurangi lebih dari 130 kali [GOY04].

Pada awalnya skema SAOTS memerlukan sekitar 5 KB untuk menyimpan setiap tanda tangan yang dibuat pada *server*. Jika pengguna dari *server* ini meningkat dengan cepat, maka tempat penyimpanan pada *server* tidak akan mampun menampung lagi setiap tanda tangan yang dibuat. Dalam hal ini dapat terlihat bahwa skema SAOTS ini sulit diimplementasikan untuk sistem komersial dan berskala besar. Terlebih lagi, perangkat *mobile* pada skema ini memerlukan sekitar 2,7 KB pada memorinya untuk menyimpan data rahasia. Untuk sebuah perangkat yang tidak dapat membuat atau memverifikasi tanda tangan, penyimpanan data ini di dalam memori sebaiknya dihindari untuk diimplementasikan.

Tetapi skema SAOTS telah dikembangkan untuk mengurangi keperluan tempat penyimpanan pada *server* dan juga penggunaan memori pada perangkat *mobile client*. Dengan menggunakan teknik baru untuk membangkitkan tanda tangan dan skema penyimpanan baru pada *server*, skema baru ini dapat mengurangi tempat penyimpanan pada *server* dan juga penggunaan memori pada perangkat *mobile client*. Hanya diperlukan 60 *byte* untuk menyimpan setiap tanda tangan yang dibangkitkan pada *server* dibandingkan dengan 5 KB pada skema sebelumnya. Kemudian penggunaan memori pada perangkat *mobile client* dikurangi dari 2,7 KB menjadi 20 byte. Kebutuhuan komputasi pada skema baru ini tetap sama dengan skema sebelumnya.

Penjelasan skem SAOTS yang baru memerlukan beberapa pengenalan notasi yang digunakan. Notasi-notasi tersebut sebagai berikut:

1. U : Perangkat *mobile client* atau pengguna.
2. VS : *Verifiable Server*.
3. R : Penerima tanda tangan.
4. L : Panjang keluaran fungsi *hash* satu arah.
5. m : Jumlah komponen kunci publik/ privat yang digunakan pada skema *one time signature*. Jumlah komponen ini sebesar $L+\log_2(L)$.
6. p : Jumlah rata-rata komponen pada sebuah *one time signature*. Biasanya sebesar to m/2.
7. P_U^i : Kunci publik *one time* ke-i dari pengguna U. Nilainya sama dengan koleksi dari komponen kunci publik m.
8. S_U^i : Kunci privat *one time* ke-i dari pengguna U. Nilainya sama dengan koleksi dari komponen kunci privat m.
9. $S_U^i(M)$: Pesan M yang ditandatangani dengan kunci privat *one time* S_U^i. Nilainya sama dengan koleksi dari komponen kunci privat yang relevan dan diperlukan untuk menandatangani M.
10. P_U : Kunci publik tradisional dari pengguna U.
11. $S_U(M)$: Pesan M yang ditandatangani dengan kunci publik tradisional dari pengguna U.

Untuk menginisialisasi skema ini maka pengguna aplikasi *mobile* U menginisiasi sebuah nilai *counter* i = 1 dan membangkitkan:

1. Sebuah kunci privat K.
2. Pasangan kunci one time sebagai berikut:
   $S_U^i = \{h(K,i,1), h(K,i,2), ..., h(K,i,m)\}$
   $P_U^i = \{h^2(K,i,1), h^2(K,i,2), ..., h^2(K,i,m)\}$

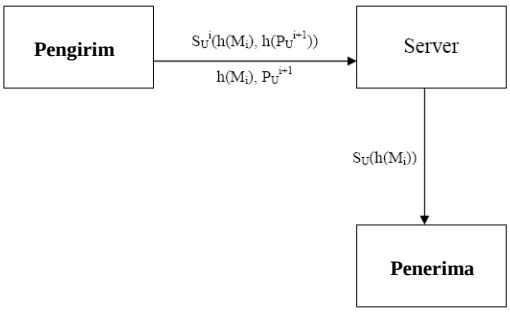
Kemudian pengguna menyimpan K dan i dan mengirimkan P_U^1 kepada *server* dengan cara non-repudiable. Hal ini dapat dilakukan dengan menggunakan sebuah tanda tangan kunci publik, jika U mempunyai kapabilitas tanda tangan tradisional atau dengan mendapatkan sertifikasi untuk kunci publik *one time* P_U^1.

$$U \rightarrow VS: P_U^1$$

Sebagai tambahan, untuk membangkitkan tanda tangan kunci publik tradisional untuk pihak pengguna, *server* membangkitkan pasangan kunci

publik dan privat, yaitu P_U dan S_U. S_U tidak akan diberitahukan pada pengguna U.

Proses pembangkitan tanda tangan dapat dilihat pada Gambar 7.



Gambar 7. Proses Pembangkitan Tanda Tangan Skema SAOTS

Untuk pembangkitan tanda tangan ke-i maka perlu dilakukan langkah-langkah berikut:

1. Pengguna U melakukan prakomptasi untuk pasangan kunci publik dan privat berikutnya (ke-i+1). Ketika pesan telah siap untuk ditandatangani, pengguna menggabungkan nilai *hash* dari pesan dengan nilai *hash* dari hasil komputasi kunci publik *one time* dan menandatangani hasilnya dengan kunci privat *one time* ke-i. Pengguna kemudian menigirimkan tanda tangan bersama dengan nilai *hash* dari pesan dan kunci publik ke *server*.
   $U \rightarrow VS: S_U^i(h(M_i), h(P_U^{i+1})), h(M_i), P_U^{i+1}$

2. *Server* kemudian memeriksa kevalidan tanda tangan yang diterimanya dengan menggunakn P_U^i yang telah disimpan. Server kemudian menyimpan $h(M_i)$, $h(S_U^i(h(M_i), h(P_U^{i+1})))$, dan $h(P_U^i)$. *Server* kemudian menimpa P_U^i dengan P_U^{i+1}. Kemudian *server* membangkitkan tanda tangan kunci publik tradisional untuk menandatangani $h(M_i)$ dengan menggunakan kunci privat pengguna S_U. Tanda tangan ini kemudian akan lansung dikirimkan kepada penerima R jika telah ditentukan oleh pengguna terlebih dahulu. Jika tidak ditentukan oleh pengguna, maka tanda tangan akan dikembalikan kepada pengguna.
   $VS \rightarrow R: S_U(h(M_i))$

Kedua langkah di atas merupakan langkah-langkah yang diperlukan dalam penggunaan skema SAOTS yang baru. Skema ini sangat transparan kepada penerima karena tanda tangan yang dibangkitkan merupakan tanda tangan standar. Sertifikasi dari pengguna P_U dapat digunakan bersamaan dengan tanda tangan jika diperlukan. Pihak penerima tidak perlu untuk mempunyai perangkat lunak baru untuk mengecek atau autentikasi tanda tangan tersebut. Kemudian pengguna tidak perlu melakukan operasi kunci publik apapun.

Perbandingan dari skema SAOTS yang lama dan baru dapat dilihat pada Tabel 1.

	Pihak	SAOTS	SAOTS baru
Kebutuhan Komputasi	Pengguna	1H + 1E	1H + 1E
	Server	1E + (p+2)H + 1S	1E + (p+3)H + 1S
	Penerima	1H + 1V	1H + 1V
Kebutuhan Penyimpanan	Pengguna	mH	1K + 1C
	Server	(m+p+1)H	3H
	Penerima	-	-

Tabel 1. Tabel Perbandingan Skema SAOTS Lama dan Baru

Keterangan:
H : komputasi *hash*
S : tanda tangan kunci publik tradisional
V : verifikasi tanda tangan kunci publik tradisional
E : komputasi *one time signature* (kurang dari satu kali proses *hash*)
p : jumlah komputasi hash untuk verifikasi one time signature
m : jumlah komponen kunci publik pada skema *one time signature*
K : ukuran dari kunci privat pengguna
C : ukuran dari counter tanda tangan

6.2 Skema Tanda Tangan Dijital Berbasiskan Client dengan Kartu SIM

Penggunaan sebuah *smart card* yang dapat mendukung fungsionalitas telepon dan juga tanda tangan dijital merupakan sebuah solusi yang baik untuk proses penandatanganan. Pengguna dapat menandatangani sebuah dokumen dan mendistribusikannya melalui layanan yang tersedia pada perangkat *mobile* miliknya misalnya GPRS atau UMTS [ROS04].

Untuk memberikan tingkat keamanan yang lebih baik lagi maka diperlukan sebuah kontrol akses yang dapat dipercaya untuk mengakses fungsionalitas tanda tangan dijital ini. Penggunaan PIN saja untuk mengkontrol akses fungsionalitas telepon tidaklah cukup karena pengguna biasanya selalu membiarkan perangkat *mobile* dan kartu SIM tidak terkunci. Seperti kartu tanda tangan tradisional, kartu SIM pun dapat disertifikasi berdasarkan kriteria evaluasi keamanan dan berada di bawah kendali pengguna perangkat.

Walaupun begitu penggunaan sebuah *smart card* untuk berbagai macam tujuan memunculkan berbagai macam persoalan. Kartu SIM dikeluarkan oleh penyedia layanan telekomunikasi sedangkan SSCD (*secure signature creation device*) dikeluarkan melalui sertifikasi penyedia layanan.

Mengkombinasikan kedua fungsi di dalam sebuah kartu menimbulkan persoalan tentang pihak yang berhak mempunyai kendali atas kunci dan sertifikasinya.

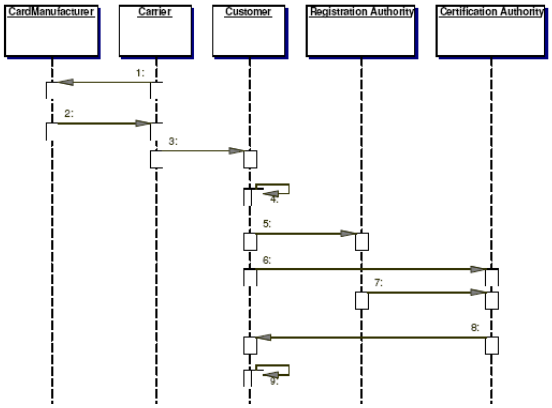
Solusi yang mudah adalah dengan menjadikan *carrier* juga menginisialisasi kerahasiaan tanda tangan untuk bertindak sebagai penyedia layanan yang dapat dipercaya oleh penggunanya. Walaupun begitu hal ini juga tetap menimbulkan permasalahan. Permasalahan pertama adalah setiap pengguna ingin agar proses berlangsung segera dan langsung tersimpan pada kartu SIM, sehingga perangkat *mobile*-nya dapat digunakan daripada harus menunggu beberapa minggu untuk penyelesaian prosedur sertifikasi. Terlebih lagi pengaitan kunci kepada *carrier* akan mengakibatkan timbulnya kesulitan bagi pengguna untuk berpindah ke *carrier* lainnya di masa yang akan datang. Bagi pihak *carrier*, hal ini memberikan keuntungan sedangkan pihak pengguna menginginkan kemungkinan untuk dapat memilih antar penyeida layanan sertifikasi dengan mudah.

Terlebih lagi karena kurangnya kesuksesan pada pasar tanda tangan dijital ini maka para penyedia layanan mungkin tidak ingin untuk menanamkan investasinya dalam pembangunan dan pengaturan pusat pelayanan sertifikasi. Sebagai tambahan, pihak penyedia layanan tidak ingin untuk mengubah jalur distribusinya kecuali jika diperlukan peningkatan pemasukan. Karena itulah diperlukan sebuah solusi yang berbeda untuk tanda tangan dijital aplikasi *mobile* dan sertifikasi, yang memungkinkan pemisahan antara informasi pendaftaran dan layanan sertifikasi.

Solusi yang diperlukan ini disebut sertifikasi berdasarkan permintaan (*certification on demand*). Operator *mobile* dapat menjual kartu-kartu SIM yang dilengkapi dengan sebuah pembangkit kunci untuk satu atau lebih pasangan kunci yang dapat digunakan untuk fungsionalitas tanda tangan. Setelah mendapatkan kartu SIM dari pihak operator, pelanggan kemudian dapat membangkitkan kunci dan mengaktivasi komponen tanda tangan dan kunci publik yang dapat disertifikasi oleh pihak penyedia layanan berdasarkan permintaan pelanggan.

Melalui pemisahan fungsionalitas telepon dan sertifikasi dari identitas pengguna oleh sebuah penyedia layanan sertifikasi, kedua fungsi tersebut dapat dijual secara terpisah dan didapatkan dari penyedia layanan yang berbeda. Pihak carrier kemungkinan akan menghadapi peningkatan biaya dalam pembuatan kartu SIM yang memungkinkan proses tanda tangan dijital tetapi juga dapat mengharapkan meningkatknya arus komunikasi dengan adanya layanan tanda tangan ini.

Langkah-langkah dalam distribusi kartu SIM dan proses sertifikasi dapat dilihat pada Gambar 8.



Gambar 8. Skema Sertifikasi Berdasarkan Permintaan

Keterangan:

1. Carrier memberikan pasangan IMSI (*International Mobile Subscriber Identity*) dan Ki (*Individual subscriber authentication key*) kepada produsen kartu.
2. Produsen kartu kemudian mengembalikan kartu SIM yang menyimpan pasangan IMSI dan Ki, sebuah pembangkit kunci untuk aplikasi tanda tangan dijital, dan kunci publik *carrier*.
3. Kartu SIM kemudian dijual kepada pelanggan dan carrier menyediakan fungsi yang digunakan untuk membangkitkan kunci dan mengaktifkan fungsionalitas tanda tangan dijital.
4. Pelanggan kemudian membangkitkan kunci dan mengaktifkan fungsionalitas tanda tangan dijital.
5. Pelanggan mendaftar pada pihak yang pendaftaran berdasarkan pilihannya, yang menyediakan informasi identifikasi dan kunci publiknya.
6. Pelanggan mengirimkan informasi identifikasinya yang ditandatangani dengan kunci privatnya dan dikirimkan kepada pihak penyedia sertifikasi.
7. Pihak pendaftaran mengirimkan kunci publik dan informasi identifikasi kepada pihak penyedia sertifikasi.
8. Jika informasi yang diberikan oleh pelanggan dan pihak pendaftaran sama, maka pihak penyedia sertifikasi memberikan sertifikat kepada pelanggan dengan mengirimkannya kepada perangkat mobile yang dimiliki pengguna.
9. Pengguna kemudian dapat memverifikasi keabsahan sertifikatnya dengan mengecek sertifikat yang dikeluarkan oleh penyedia layanan sertifikasi.

Langkah-langkah di atas menunjukkan bahwa tidak perlu adanya perubahan pada infrastruktur distribusi operator *mobile*. Langkah 1 sampai 3 tetap sama seperti sebelumnya, terpisah dari fakta bahwa produsen kartu menambahkan informasi dan fungsionalitas tambahan pada kartu SIM. Untuk memastikan bahwa produsen kartu tidak mengetahui kuci pricat dari pengguna, maka pembangkitan kunci harus dilakukan oleh kartu SIM yang bersangkutan. Pelanggan tidak dipaksa untuk mensertifikasikan kuncinya. Pelanggan dapat hanya menggunakan fungsionalitas telpon saja. Pelanggan juga dapat mengaktifkan fungsionalitas tanda tangan tanpa melalui prosedur sertifikasi. Jika pelanggan menginginkan tanda tangan dijital yang legal, maka harus dilalui prosedur sertikasi yang dibutuhkan. Pelanggan dapat memilih secara bebas penyedia layanan sertifikasi ini.

Fungsi pembangkitan kunci dan pengaktifan fungsionalitas tanda tangan dijital pada langkah 4 dapat digunakan untuk memastikan bahwa tidak ada tanda tangan yang dibuat sebelum pelanggan mempunyai kendali atas kartu SIM-nya. Jika aplikasi tanda tangan telah diaktifkan sebelumnya maka pengguna akan mengetahuinya ketika mengakses fungi ini.

Langkah 6 dapat dihilangkan tetapi menyediakan layanan kepada pelanggan untuk memastikan bahwa integritas dari informasi identifikasinya tersimpan dengan baik. Jika pengguna ingin berganti penyedia layanan sertifikasi maka hanya perlu dilalui langkah 5 hingga 9 dengan penyedia layanan sertifikasi yang baru. Jika pelanggan berganti *carrier*, maka ia perlu mengulangi keseluruhan prosedur kembali, tetapi dapat mendaftar kembali kepada penyedia layanan sertifikasinya yang sedang digunakan oleh pelanggan tersebut.

Penggunaan kartu SIM ini mengakibatkan perangkat *mobile* tidak perlu melakukan komputasi pembangkiatan kunci untuk tanda tangan dijital. Pembangkitan kunci dilakukan oleh pihak penyedia layanan atau dapat dilakukan sendiri yang kemudian akan tersimpan pada kartu SIM. Untuk melakukan proses tanda tangan dijital, maka pengguna hanya perlu menggunakan kunci yang telah tersedia pada kartu SIM. Setiap pemberian tanda tangan akan selalu menggunakan kunci yang sama selama kunci pada kartu SIM tidak berubah.

Dengan cara ini maka perangkat *mobile* hanya perlu menyediakan sedikit memori untuk menyimpan kuncinya dan melakukan sedikiti komputasi untuk melakukan proses tanda tangan dijital. Perangkat mobile tidak perlu melakukan proses pembangkitan kunci setiap kali akan memberikan tanda tangan dijital.

6.3 Tanda Tangan Dijital Berbasiskan Server (Server Based Signature)

Sebuah layanan *server based signature* (SBS) mempunyai kelas-kelas layanan non-repudiation. Non-Reoudiation of Sender (NRS) memastikan bahwa pengirim pesan tidak dapat menyangkal telah mengirimkan pesan dan Non-Reoudiation of Receiver (NRR) menjamin bahwa penerima pesan tidak dapat menyangkal telah menerima pesan yang dimaksud [LEI04].

Skema dan arsitektur SBS untuk kelas NRS dimulai dengan persetujuan setiap pihak yang terlibah dalam penggunaan fungsi *hash* satu arah yang bebas kolisi, misalnya dengan menggunakan SHA1 dan MD5. Kemudian pengirim dan penerima harus mengambil fungsi *hash* pribadi sebagai pasangan $h_{snd}()$ dan $h_{rev}()$. Kedua fungsi ini disebut sebagai fungsi hash dengan kunci. Hal ini dapat dengan mudah dilakukan dengan memasukkan identitas pengirim atau pengerima sebagai argumen dari fungsi *hash* ini. $h^i_{snd}()$ dan $h^i_{rcv}()$ menunjukkan fungsi *hash* dengan kunci yang terkait dengan identitas berindeks i (i kali operasi *hash*).

Setiap pengguna membangkitkan kunci pricat K_u (u merepresentasikan pengguna). Dengan menggunakan K_u sebagai input awal, pengguna dapat membangun satu rantai fungi *hash* K^0_u, K^1_u, ..., K^n_u, dengan $K^0_u = K_u$ dan $K^i_u = h^i_u(K^{i-1}_u)$. $PK_U = K^n_u$ dianggap sebagai kunci publik dari pengguna. Dengan asumsi pengirim ingin mengirimkan pesan x kepada penerima, maka proses tanda tangan dijital diilustrasikan sebagai berikut.

Langkah pertama yang perlu dilakukan adalah pengirim membangkitkan kunci publiknya PK_{snd}. Setelah itu pengirim memilih *server* untuk tanda tangan (S). Kemudian pengirim mengirimkan identitas pengguna, jumlah maksimum tanda tangan n, kunci public dasar PK_{snd} dan S kepada sebuah *Certification Authorities* (CA) untuk mendapatkan sebuah sertifikasi. Bentuk sertifikasi adalah sebagai berikut:

$$Cert_{snd} = SK_{CA}(Sender, n, PK_{snd}, S)$$

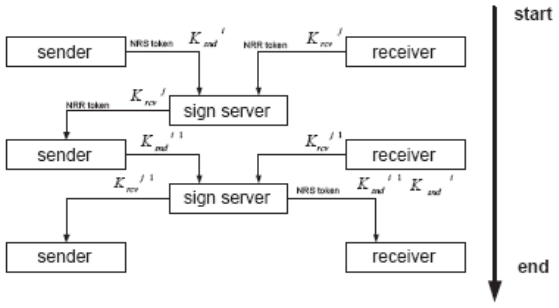
SK_{CA} menunjukkan kunci privat CA. Setelah proses sertifikasi diselesaikan, hasilnya dapat dipublikasikan melalui layanan *directory*, misalnya LDAP.

Langkah berikutnya adalah *server S* membangkitkan dua buah bilangan prima besar p dan q. Kemudian menghitung nilai $m = p \times q$, $\phi(m) = (p-1) \times (q-1)$. Sebuah angka d juga dihitung dengan cara $e \times d = 1 \bmod \phi(m)$ dengan nilai $e = 3$. S kemudian menyimpan (d, p, q) sebagai kunci pricat dan mempublikasikan (e, m).

Langkah ketiga dilakukan untuk pembangkitan tanda tangan dijital. Jika pengirim ingin membangkitkan sebuah tanda tangan dijital untuk sebuah pesan x, pertama-tama pengirim mimilih secara acak dua buah bilangan r dan v, dengan $r, v \in Z^*_n$. Kemudian dihitung nilai $\delta = r^e h(x)(v^2 + 1) \bmod m$ dan mengirimkan (δ, K^i_{snd}, i) kepada S. Kemudian S secara acak memilih sebuah bilangan positif z, $z < m$ dan mengirimkannya kembali kepada pengirim. Setelah menerima z, pengirim memilih secara acak bilangan bulat lainnya r' dan menghitung $b = r \, x \, r'$. Kemudian pengirim menghitung $\acute{\eta} = b^e \, x \, (v - z)$ dan mengirimkannya kepada S.

Langkah keempat dilakukan dengan S menghitung nilai $\gamma = \acute{\eta}^{-1} \bmod m$ dan
$t = h(Cert_{snd})^d(\delta(z^2 + 1) \, \acute{\eta}^{-2})^{2d} \bmod m$, kemudian mengirimkan (γ, t) kepada pengirim. Setelah menerima (γ, t), pengirim menghitung nilai
$c = (vz + 1) \, x \, \gamma \, x \, b^e = (vz + 1)(v - z)^{-1} \bmod n$, dan
$s = t \text{ x } r^2 \text{ x } r^4 \bmod m$. Ketiga $(Cert_{snd}, c, s)$ adalah tanda tangan dari pesan x. Karena
$s^e \equiv h(Cert_{snd})h(x)^2(c^2 + 1)^2 \bmod m$, sehingga setiap pihak dapat memverifikasi tanda tangan $(Cert_{snd}, c, s)$ terhadap pesan x dengan mudah dan efisien. Setelah memverifikasi tanda tangan, pengirim, mengirimkan $(Cert_{snd}, c, s)$ dan $K^{i-1}{}_{snd}$ kembali kepada *server*. S kemudian memverifikasi tanda tangan dan $K^i{}_{snd} = h(K^{i-1}{}_{snd})$. Jika semua hasilnya adalah benar, S mengirimkan pesan x dengan tanda tangannya kepada penerima. $(Cert_{snd}, c, s)$, $K^i{}_{snd}$, dan $K^{i-1}{}_{snd}$ merupakan komponen NRS.

Proses pemberian tanda tangan ini dapat dilihat pada Gambar 9.



Gambar 9. Alur dari SBS

Skema dan arsitektur SBS untuk kelas NRR diselesaikan dengan memanfaatkan pengembangan dari skema SRS. Setelah proses yang dijelaskan dari langkah 1 hingga 5, penerima juga membangkitkan tanda tangan $(Cert_{rcv}, c', s')$. $(Cert_{rcv}, c', s')$ merupakan kompenen NRR. Pada langkah 4, sebelum pengirim mengirimkan $K^{i-1}{}_{snd}$ kepada *server*, penerima harus mengirimkan komponen NRR-nya bersama dengan kunci publik $K^j{}_{rcv}$ kepada pengirim. Kemudian pengirim mengirimkan $K^{i-1}{}_{snd}$ kepada *server* setelah memverifikasi komponen NRR dari penerima. Setelah *server* menerima $K^{i-1}{}_{snd}$ dan $K^{j-1}{}_{rcv}$, tanda tangan dijital pengirim $((Cert_{snd}, c, s), i, K^i{}_{snd}, K^{i-1}{}_{snd})$ dikirimkan kepada penerima dan $K^{j-1}{}_{rcv}$ kepada pengirim.

Pada kasus pengirim memberikan klaim tidak mengirimkan pesan yang ditandatangani oleh komponen NRS, pihak penerima dapat menyediakan komponen, pesan, sertifikasi, dan tanda tangan dijital kepada pihak arbitrasi. Pihak arbitrasi pertama memverifikasi sertifikasi $(Cert_{snd})$ yang telah disertifikasi oleh CA dengan identifikasi pengirim, kunci publik, identifikasi *server*, dan lain sebagainya. Kemudian pihak arbitrasi mengecek tanda tangan $(Cert_{snd}, c, s)$ menggunakan formula $s^e \equiv h(Cert_{snd})h(x)^2(c^2 + 1)^2 \bmod m$. Terakhir, pihak arbitrasi harus menentukan kunci publik pengirim $PK_U = K^n{}_u$ dapat diturunkan dari $K^{i-1}{}_u$, $K^i{}_u$ dengan fungsi *hash* satu arah. Jika semua hasilnya adalah benar, maka pihak penengah dapat percaya bahwa pengirim telah mengirimkan pesannya kepada penerima. Dengan hasil pengecekan yang berhasil ini, jika pengirim tetap menyangkal telah mengirimkan pesan, hal ini dapat membuktikan bahwa:

1. CA melakukan kecurangan. Pada skema ini, setiap pengirim harus terlebih dahulu mendaftarkan kunci publiknya yang berasosiasi dengan *server*. Jika pengirim ingin memberikan klain bahwa CA telah melakukan kecurangan, maka CA harus memberikan sertifikasi dengan kunci publik yang berbeda. Kemudian ketika sertifikasi ini ditunjukkan, CA harus memberikan alasan terjadinya hal tersebut. CA dapat membuktikan dirinya tidak bersalah dengan menunjukkan kontrak pendaftaran yang dihasilkan pada proses sertifikasi awal.
2. *Server* melakukan kecurangan Untuk membuktikan bahwa server melakukan kecurangan, maka pengirim harus membangkitkan tanda tangan dijital lain $(Cert_{snd}, c, s)$ dan $K^{i-1}{}_{snd}$ yang dapat menandatangani pesan x. Jika pengirim tidak dapat menyediakan $K^{i-1}{}_{snd}$, sangatlah tidak mungkin secara komputasi untuk menurunkan $K^{i-1}{}_{snd}$ dari $K^i{}_{snd}$ karena karakteristik satu arah dari fungsi *hash* $h()$. Tanda tangan $(Cert_{snd}, c, s)$ kemudian dibangkikan dari pesan x dengan menggunakan SHA dan RSA, jadi pembangkitan tanda tangan palsu sangatlah tidak mungkin secara komputasi.

Seperti yang ditunjukkan pada Gambar 9, *server* yang digunakan haruslah dapat dipercaya untuk menghindari permasalahan NRR dan NRS. Seperti pada teknik kriptografi kunci asimetri, hanya dapat digunakan *verifiable server* (kecurangannya dapat dibuktikan dengan sebuah pihak arbitrasi). Permasalahan NRR mempunyai penjelasan sama seperti di atas. Bergantung pada karakteristik satu arah dan bebas kolisi dari fungsi *hash*, sangatlah

mudah untuk menghindari *repudiation* baik dari pihak pengirim pesan maupun dari pihak penerima pesan.

Biaya keseluruhan dari skema SBS ini dapat dibagi ke dalam dua buah bagian, yaitu:

1. Biaya komputasi. Dalam skema ini komputasi terdiri dari:
   a. Komputasi *server*: *Server* pertama-tama harus membangkitkan sepasang kunci RSA, kemudian mempublikasikannya melalui layanan *directory* yang aman. Terakhir, server menggunakan algoritma kriptografi kunci publik untuk menghasilkan tanda tangan dijital pada pesan *x* yang diinginkan.
   b. Komputasi pengguna: Pengguna pertama-tama harus melakukan komputasi pada nilai *hash* yang berantai dengan masukan *pre-image* khusus. Dan pada langkah 4 skema ini, pengguna harus memverifikasi tanda tangan yang dihasilkan oleh *server*.
2. Biaya komunikasi. Untuk menandatangani sebuah pesan, pengguna harus bergantung pada *server* untuk mengeksekusi sebagian besar komputasi dalam pembangkitan tanda tangan dijital. Karena itulah diperlukan jalur komunikasi yang menjembatani pengguna dan *server* selama proses pemberian tanda tangan berlangsung.

Dibandingkan dengan skema tanda tangan dijital tradisional, *overhead* dari komunikasi jaringan, pembangkitan kunci publik dasar, dan verifikasi komponen tanda tangan pada skema ini cukup besar. Walaupun begitu skema SBS ini sangat cocok untuk perangkat *mobile* yang mempunyai energi yang sangat terbatas untuk menghasilkan tanda tangan dijital yang aman. Pada skema ini, pengguna, yang menggunakan perangkat *mobile*, hanya perlu melakukan komputasi nilai *hash* yang berantai dan memverifikasi tanda tangan.

Untuk melakukan proses dekripsi sebanyak *km* bit pesan dengan menggunakan kunci sepanjang *k* bit, algoritma RSA *m* kali lebih kompleks daripada melakukan komputasi sebuah operasi *k* bit modulus eksponensial. Dengan sebuah operasi modulus *n*, komputasi untuk modulus eksponensial memerlukan $0{,}3246|n|^1$ perkalian modulus. Seperti yang telah diperlihatkan di atas, skema dan algoritma SBS membangkitkan tanda tangan dijital dengan menggunakan bebearapa operasi perkalian modulus (sekitar 20 operasi). Dengan panjang kunci normal 1024, sangatlah jelash bahwa *Server Bases Signature* menghabiskan energi lebih seidikit daripada *Server Supported Signature*.

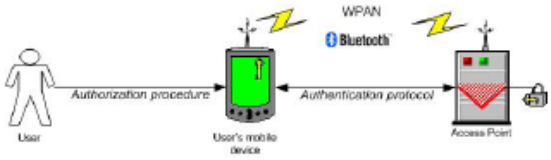
Skema SBS ini dapat digunakan dalam berbagai macam sistem, misalnya verifikasi kode *mobile*, *mobile payment*, pembelajaran jarak jauh, dan autentikasi citra. Kesemua hal ini didukung dengan adanya integrasi antara sistem autentikasi dan pertukaran di dalam skema SBS. Misalnya Alice mempublikasikan karya seni dijitalnya pada sebuah *artwork web server*. Untuk melindungi hasil karyanya maka produk-produk dijital ini diberikan teknik-teknik perlindungan seperti *watermark*. Seorang konsumen, Bob, pertam–tama melihat-lihat keseluruhan produk yang telah dipublikasikan pada *web server*. Jika Bob tertarik pada produk tertentu dan tertarik untuk membelinya, pertama ia harus membayar terlebih dahulu untuk produk tersebut melalui sebuah *server* pembayaran. Setelah Bob menyelesaikan transaksi pembayaran ini, Alice mengecek pembayaran yang dilakukan oleh Bob tersebut. Jika proses pembayaran tidak bermasalah, maka Alice memberikan tanda tangan pada *artwork* dijital yang diinginkan Bob dengan menggunakan skema SBS, kemudian mengirimkannya kepada Bob melalui *server* tanda tangan.

Skema SBS merupukan skema pemberian tanda tangan dijital untuk digunakan pada aplikasi *mobile*, termasuk palmtop, telepon selular, dan PDA. Karena lemahnya kemampuan komputasi dan terbatasnya energi yang dimiliki, maka perangkat-perangkat tersebut tidaklah cocok untuk menjalankan algoritma kriptografi kunci publik tradisional. Dengan menggunakan skema SBS ini maka kegiatan komputasi pada pihak *client* yang menggunakan perangkat *mobile* dapat dikurangi hingga sesedikit mungkin dan berakibat pada konsumsi energi yang lebih sedikit. Karena itulah skema SBS dapat dengan lebih mudah diimplementasikan daripada skema tanda tangan tradisional. Skema SBS ini pun dapat digunakan pada berbagai macam sistem, misalnya verifikasi kode mobile, mobile payment, pembelajaran jarak jauh, autentikasi citra, dan lain sebagainya.

6.4 Tanda Tangan Dijital Berbasiskan Server untuk Skema Autentikasi Personal

Untuk membangun sebuah sistem autentikasi personal pada perangkat *mobile* yang kuat maka diperlukan sebuah skema dan arsitektur untuk perangkat *mobile* yang mendukung tujuan tersebut. Arsitektur ini disebut sebagai *Mobile Architecture for Strong Personal Authentication* (MASPA). MASPA didasarkan pada entitas, algoritma, dan protokol yang berbeda-beda. Entitas utama pada MASPA adalah pengguna, perangkat *mobile* yang digunakan pengguna, dan titik akses. Protokol yang digunakan pada MASPA adalah protokol komunikasi dan autentikasi. Sedangkan algoritma yang digunakan dalam arsitektur MASPA adalah algoritma-algoritma dasar kriptografi seperti fungsi *hash*, algoritma kriptografi kunci simetri, dan algoritma kriptografi kunci publik [HER02].

Proses autentikasi secara garis besar dapat dilihat pada Gambar 10.



Gambar 10. Proses Autentikasi MASPA Secara Global

Perangkat *mobile* adalah sebuah entitas yang beraksi berdasarkan penggunanya. Peran dari perangkat *mobile* sama seperti peran *smart card* pada skema autentikasi tradisional dalam hal kedua perangkat tersebut mendukung pengguna untuk menyimpan dan melakukan komputasi nilai kriptografi yang diperlukan untuk proses autentikasi.

Karakteristik dasar dari sebuah perangkat *mobile* adalah sebagai berikut:

1. Sebuah perangkat *mobile* harus menyediakan sejumlah *non-volatile* memori yang terbatas.
2. Kemampuan komputasi perangkat *mobile* harus memungkinkan untuk mengeksekusi algoritma kriptografi yang diperlukan dengan waktu yang masuk akal.
3. Perangkat *mobile* harus dapat mengirimkan data melalui *wireless personal area network* (WPAN) seperti Bluetooth.
4. Perangkat *mobile* jarus sangat portabel dalam hal pengguna harus dapat secara permanen membawa perangkat tersebut.

Karakteristik pertama dan kedua memungkinkan perangkat *mobile* untuk menyimpan informasi sehingga dapat menyeidakan lingkungan yang aman dan terpercaya sehingga pengguna dapat melakukan proses autentikasi sendiri. Dalam hal ini, informasi rahasia yang sangat penting dalam proses autentikasi tidak pernah meninggalkan perangkat *mobile*. Selain itu keamanan perangkat *mobile* juga terjamin sebagian melalui kendali fisik oleh pengguna.

Titik akses merepresentasikan entitas yang memverifikasi indentitas pengguna. Walaupun begitu proses verifikasi dapat dilakuka di luar titik akses, misalnya pada *server* autentikasi. Titik akses ini harus menyediakan fitur-fitur sebagai berikut:

1. Titik akses harus dapat menyimpan informasi yang dipertukarkan selama protokol autentikasi berlangsung.
2. Kemampuan komputasi kriptografi titik akses juga diasumsikan sangat baik dalam melakukan protokol autentikasi.
3. Titik akses juga harus menyediakan teknologi WPAN yang sama dengan perangkat *mobile* untuk proses pengiriman data antara kedua entitas.
4. Untuk aplikasi autentikasi ad-hoc, titik akses harus dapat menyimpan profil pengguna.

MASPA dirancang agar dapat menggunakan teknologi WPAN apapun untuk menghubungkan antar entitas. Walaupun begitu, untuk implementasinya, dipilih teknologi Bluetooth. Teknologi Bluetooth dipilih karena berbiaya rendah, konsumsi energi rendah, dan merupakan teknologi nirkabel berjarak pendek.

Prosedur autorisasi dilakukan oleh pengguna dan perangkat *mobile*-nya pada permulaan proses autentikasi. Tujuan dari prosedur autorisasi adalah untuk memastikan pengguna yang menggunakan perangkat *mobile* tersebut merupakan pemilik yang sebenarnya dan pemilik dari informasi rahasia pada perangkat *mobile* tersebut.

MASPA menggunakan prosedur yang berbasiskan *password*. Pengguna memasukkan kata kunci ke dalam perangkat *mobile*-nya dan kemudian diverikasi kebenaran informasinya.

Protokol autentikasi yang digunakan pada MASPA berdasarkan pada arsitektur protokol yang diusulkan oleh Diffie-Hellman. Proses autentikasi ini dapat dilihat pada Gambar 11.

$$
\begin{aligned}
M \longrightarrow A &: g^{r_m} && (1)\\
A &: K = (g^{r_m})^{r_a}\\
M \longleftarrow A &: g^{r_a}, E_K\{Sig_a(g^{r_a}, g^{r_m})\}, Cert_a && (2)\\
M &: K = (g^{r_m})^{r_a}\\
M \longrightarrow A &: E_K\{Sig_m(g^{r_a}, g^{r_m}), Cert_m\} && (3)
\end{aligned}
$$

Gambar 11. Protokol Autentikasi MASPA

Dapat diperhatikan bahwa protokol autentikasi ini memungkinkan penggunaan algoritma kriptografi yang berbeda. Sebagai contoh perangkat *mobile* dan titik akses harus dapat melakukan dan memverifikasi tanda tangan dijital (karena itu harus dapat menggunakan fungsi *hash*) dan sebuah algoritma kriptografi kunci simetri.

Proses autentikasi pada arsitektur MASPA dilakukan dengan menggunakan perangkat *mobile* dan WPAN dibandingkan dengan *smart card*. Teknologi WPAN ini mempunya kelebihan yaitu *deployment* yang cepat dan biasanya telah dapat digunakan pada beragam perangkat *mobile*. Sehingga proses autentikasi ini bergantung pada perangkat keras yang dimiliki oleh perangkat *mobile* dan titik akses. Walaupun begitu MASPA tetap membutuhkan algoritma kriptografi seperti fungsi hash, algoritma kriptografi kunci simetri, dan algoritma kriptografi kunci publik.

7 Kesimpulan

Berdasarkan hasil studi dapat disimpulkan hal-hal sebagai berikut:

1. Semakin maraknya penggunaan perangkat *mobile* dalam bidang-bidang yang penting diperlukan suatu skema pengamanan, penjagaan integritas data, dan autentikasi. Semua hal ini dapat dilakukan dengan menggunakan tanda tangan dijital.
2. Perangkat *mobile* mempunyai keterbatasan memori, energi, dan kemampuan komputasi. Sedangkan algoritma tanda tangan dijital dengan fungsi *hash* yang banyak digunakan saat ini membutuhkan memori yang cukup besar dan kemampuan komputasi yang tinggi (sehingga memerlukan energi yang cukup besar juga). Untuk mengatasi permasalahan ini diperlukan protokol, skema, dan arsitektur tanda tangan dijital untuk perangkat *mobile* dengan memperhatikan keterbatasan yang ada.
3. Protokol, skema, dan arsitektur tanda tangan dijital untuk perangkat *mobile* dapat digunakan juga untuk memastikan proses keamanan, penjagaan integritas data, dan autentikasi berlangsung dengan baik.
4. Protokol, skema, dan arsitektur tanda tangan dijital untuk perangkat *mobile* berbasiskan *client* digunakan dengan cara melakukan prakomputasi untuk pembangkitan kunci dan parameter lain yang dibutuhkan. Semua parameter yang diperlukan disimpan pada perangkat *mobile* untuk kemudian digunakan. Hal ini berakibat pada sulitnya untuk mengikuti perubahan teknologi yang digunakan.
5. Protokol, skema, dan arsitektur tanda tangan dijital untuk perangkat *mobile* berbasiskan *server* digunakan dengan adanya kolaborasi antara *client* dan *server*. Komputasi kriptografi dititikberatkan pada sisi *server* karena adanya keterbatasan pada perangkat *client*. Hal ini berakibat pada mudahnya beradapatasi terhadap segala perubahan yang terjadi.
6. Penggunaan teknologi terkini misalnya teknologi jaringan nirkabel personal seperti Bluetooth dapat memacu perkembangan protokol, skema, dan arsitektur tanda tangan dijital untuk perangkat *mobile*.

8 Daftar Referensi

[ELB02] Elbaz, Limor. 2002. *Using Public Key Cryptography in Mobile Phones*.

[FRI03] Fritsch, Lothar; Ranke, Johannes; Rossnagel, Heiko. 2003. *Qualified mobile electronic signatures: Possible, but worth a try?*

[GOY04] Goyal, Vipul. 2004. *More Efficient Server Assisted One Time Signature*.

[HER02] Herrera, Jordi Joancomart´i; Prieto, Josep Bl´azquez. 2002. *A Personal Authentication Scheme Using Mobile Technology*.

[JAK01] Jakobsson, Markus; Pointcheval, David. 2001. *Mutual Authentication for Low-Power Mobile Devices*.

[LEI04] Lei, Yu; Chen, Deren; Jiang, Zhongding. 2004. *Generating Digital Signatures on Mobile Devices*.

[LIN02] Lin, Binshan. 2002. *Wireless Technology Infrastcture and Decisions*.

[MUN06] Munir, Rinaldi. 2006. *Diktat Kuliah IF5054 Kriptografi*. Program Studi Teknik Informatika - Institut Teknologi Bandung.

[ROS04] Rossnagel, Heiko. 2004. *Mobile Qualified Electronic Signatures for Secure Mobile Brokerage*.

[SCH96] Schneier, Bruce. 1996. *Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C*. John Wiley and Sons.