

Laporan Tugas Akhir Kuliah
EL-695 Keamanan Sistem
Dosen: **Dr. Ir. Budi Rahardjo**

Metoda Pemberian *Signature* Pada Sinyal Telepon

Oleh :

Nama : Theodorus Eric

Nim : 232 00 062



PROGRAM PASCA SARJANA
JURUSAN TEKNIK ELEKTRO
INSTITUT TEKNOLOGI BANDUNG
2002

Daftar Isi

| | halaman |
|--|---------|
| Daftar Isi | 1 |
| Daftar Gambar dan Tabel | 2 |
| Abstrak | 3 |
| 1. Pendahuluan | 3 |
| 2. Tinjauan Sistem Verifikasi Pada Telepon | 5 |
| 2.1 Sekilas Mengenai Digital <i>Signature</i> | 5 |
| 2.2 Permasalahan Dalam Pemberian <i>Signature</i> Pada Sinyal Telepon | 6 |
| 3. Pemberian <i>Signature</i> Pada Sinyal Telepon | 7 |
| 3.1 Fungsi <i>Hash</i> Untuk Sinyal Telepon | 7 |
| 3.2 Enkripsi Keluaran Fungsi <i>Hash</i> | 10 |
| 3.3 Penyisipan <i>Signature</i> Menggunakan Teknik <i>Watermarking</i> | 10 |
| 4. Verifikasi <i>Signature</i> | 12 |
| 4.1 Verifikasi Pada Percakapan Telepon | 12 |
| 4.1.1 Proses <i>Hash</i> | 13 |
| 4.1.2 Pendeteksian <i>Signature</i> | 13 |
| 4.1.3 Dekripsi <i>Watermark</i> | 14 |
| 4.2 Verifikasi <i>Signature</i> Pada Potongan Rekaman | 14 |
| 5. Hasil Percobaan | 15 |
| 5.1 Hasil Percobaan Untuk Fungsi <i>Hash</i> | 15 |
| 5.2 Hasil Percobaan Untuk <i>Watermark</i> | 16 |
| 6. Upaya Pemalsuan | 17 |
| 6.1 Cara Pemalsuan | 17 |
| 6.2 Keamanan Terhadap Upaya Pemalsuan | 18 |
| 7. Kesimpulan | 19 |
| Referensi | 20 |
| Appendix : <i>Source Code Matlab</i> | 21 |
| Pemberian <i>Signature</i> | 21 |
| Verifikasi <i>Signature</i> | 23 |

Daftar Gambar dan Tabel

| | halaman |
|---|---------|
| Daftar Gambar | |
| Gambar 2.1 Sistem verifikasi penelepon menggunakan sistem kunci publik | 5 |
| Gambar 2.2 Digital <i>signature</i> dengan fungsi <i>Hash</i> | 6 |
| Gambar 3.1 Pemberian <i>signature</i> pada sinyal telepon | 7 |
| Gambar 3.2 Contoh sinyal suara | 9 |
| Gambar 3.3 Contoh pola sinyal <i>pseudo noise</i> yang digunakan dalam percobaan dan hasil proyeksinya terhadap blok sinyal suara | 9 |
| Gambar 3.4. Contoh sinyal <i>pseudo noise</i> untuk <i>watermark</i> | 11 |
| Gambar 3.5 <i>Watermark</i> menggunakan sinyal <i>pseudo noise</i> | 11 |
| Gambar 3.6 Sinyal asli dan sinyal yang telah diberi <i>watermark</i> | 12 |
| Gambar 4.1 Proses verifikasi <i>signature</i> | 12 |
| Gambar 4.2 Pedeteksian bit <i>watermark</i> | 14 |
| Daftar Tabel | |
| Tabel 3.1 Hasil proyeksi blok sinyal suara dengan 15 pola <i>pseudo noise</i> | 9 |

Metoda Pemberian *Signature* Pada Sinyal Telepon

Theodorus Eric
Nim: 232 00 062
Jurusan Teknik Elektro
Institut Teknologi Bandung, Indonesia
eric@hakkinen.com

Abstrak

Laporan ini mengusulkan suatu metoda pemberian *signature* untuk verifikasi penelepon dengan sistem kunci publik. *Signature* yang diberikan berasal dari keluaran fungsi *Hash* khusus yang dienkripsi menggunakan *RSA*. Sinyal telepon yang analog memerlukan suatu fungsi *Hash* khusus yang kuat (*robust*) karena sinyal telepon sering mengalami gangguan seperti adanya *noise* atau sambungan yang terputus-putus. Fungsi *Hash* yang digunakan untuk sinyal telepon ini bekerja dengan cara memproyeksikan N buah pola sinyal *pseudo noise* ke sinyal telepon untuk menghasilkan keluaran *Hash*. *Signature* tersebut disisipkan ke dalam sinyal percakapan sebagai *watermark* untuk diverifikasi di penerima. Pada laporan ini juga dibahas mengenai kekuatan *signature* yang digunakan dan kemungkinan terjadinya pemalsuan suara.

1. Pendahuluan

Verifikasi terhadap penelepon perlu dilakukan untuk mencegah terjadinya pemalsuan atau penipuan. Penipuan melalui telepon sudah banyak terjadi misalnya saja seperti yang terjadi di Bali dimana seseorang yang mengaku dari PT. Telkom menelepon seseorang untuk memberitahu kalau orang tersebut telah memenangkan hadiah mobil dan kemudian meminta untuk mentransfer uang ke rekening tertentu [1]. Di Bandung dan Jakarta terjadi kasus dokter gadungan yang menelepon untuk meminta ongkos perawatan

rumah sakit [2]. Ada lagi yang menelepon dengan mencatut nama pejabat untuk meminta transfer uang [3].

Permasalahannya adalah kita tidak dapat mengetahui adalah apakah betul telepon tersebut berasal dari bank yang bersangkutan. Dalam hal ini kita memerlukan suatu cara untuk melakukan verifikasi terhadap penelepon.

Kasus lain misalnya A tidak mempercayai B. Setelah B menerima telepon dari A, kemudian B membuat rekaman percakapan palsu. Bagaimana cara membuktikan bahwa percakapan tersebut memang benar terjadi. Dalam hal ini kita memerlukan suatu cara untuk dapat melakukan verifikasi terhadap rekaman tersebut.

Verifikasi terhadap penelepon dapat dilakukan menggunakan sistem *caller ID*. Sistem ini mengirimkan informasi tambahan berupa identitas penelepon sebelum percakapan dimulai [4], namun cara ini tidak aman karena mungkin saja seseorang mengubah informasi *caller ID* tersebut. Dalam hal ini sebaiknya kita tidak hanya mengirimkan identitas penelepon sebelum pembicaraan berlangsung, tapi kita juga perlu memberikan suatu *signature* yang berhubungan dengan pesan yang kita kirim seperti pada *teknik digital signature* [5].

Teknik *digital signature* menggunakan fungsi *Hash* dengan sistem kunci publik (misalnya *RSA*) bersifat sensitif sehingga tidak dapat diterapkan secara langsung pada sinyal telepon karena sinyal telepon yang analog sering mengalami gangguan seperti adanya *noise* atau sambungan yang terputus-putus. Fungsi *Hash* standard seperti MD5 dan SHA-1 tidak dapat digunakan untuk sinyal telepon karena keluarannya akan berubah secara dramatis karena sedikit perubahan pada masukannya [6]. Untuk sinyal telepon, fungsi *Hash* yang digunakan harus menghasilkan keluaran yang sama untuk sinyal suara dengan adanya sedikit gangguan, namun menghasilkan keluaran yang berbeda untuk sinyal suara yang berbeda. Dalam hal ini kita memerlukan suatu fungsi *Hash* khusus untuk mendapatkan *digest message* pada sinyal telepon.

Untuk itu perlu dicari suatu metoda yang tepat untuk memberikan *signature* pada sinyal telepon. Laporan ini mengusulkan suatu metoda pemberian *signature* pada sinyal telepon yang tahan terhadap gangguan dan dapat diverifikasi dengan menggunakan sistem kunci publik.

2. Tinjauan Sistem Verifikasi Pada Telepon

Sistem ini bekerja melakukan verifikasi (lihat Gambar 2.1) seperti pada teknik *digital signature* namun berbeda pada cara pengiriman *signature* dan fungsi *Hash* yang digunakan karena di sini digunakan sinyal telepon yang analog. Proses verifikasi tersebut dapat terjadi dua arah selama percakapan dimana B melakukan verifikasi terhadap A dan A juga melakukan verifikasi terhadap B.



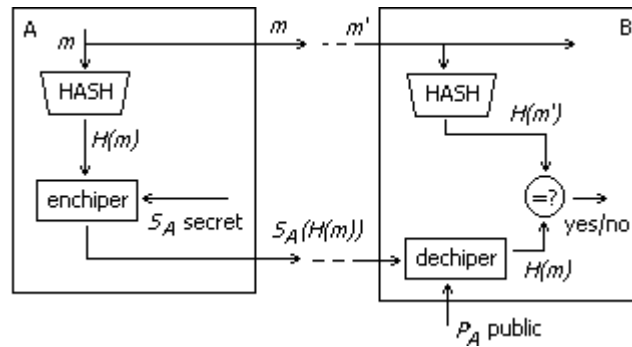
Gambar 2.1 Sistem verifikasi penelepon menggunakan sistem kunci publik

Berikut ini akan dibahas sekilas mengenai *digital signature* dan selanjutnya akan dibahas mengenai pemberian *signature* pada sinyal telepon.

2.1 Sekilas Mengenai *Digital Signature*

Pada *digital signature*, semua orang dapat men-*dechiper* data, namun hanya orang yang berwenang (*authorised transmitter*) yang dapat men-*enchiper* pesan [5]. Hanya orang yang berwenang yang memiliki akses ke kunci rahasia S_A dan tidak mungkin untuk mendapatkan S_A dari kunci publik P_A . *Chipertext* C dapat dijadikan sebagai jaminan bahwa pesan tersebut dikirim oleh orang yang berwenang (*authorised person*). Tidak ada orang lain yang dapat membangkitkan C karena hanya orang yang berwenang (*authorised person*) saja yang mengetahui S_A . Jika S_A lain digunakan, maka pesan tidak dapat di-*dechipe*.

Untuk mendapatkan *digital signature*, seringkali data mengalami pemrosesan awal (*pre-processed*), sebagai contoh dengan fungsi *Hash*. Bila autentifikasi dilakukan menggunakan algoritma *RSA* maka ini dapat meningkatkan efisiensi mengingat kompleksitas dari algoritma *RSA* menuntut data yang kompak (*compact*) [5]. Gambar 2.2 di bawah ini menunjukkan sistem *digital signature* menggunakan fungsi *Hash* [5]:



Gambar 2.2 Digital *signature* dengan fungsi *Hash*

2.2 Permasalahan Dalam Pemberian *Signature* Pada Sinyal Telepon

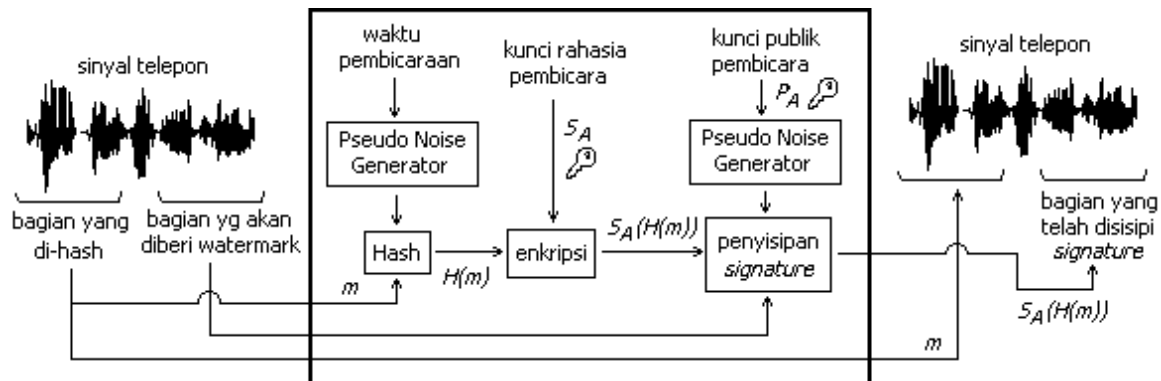
Pemberian *signature* sinyal telepon menimbulkan beberapa masalah yaitu dalam hal penggunaan fungsi *Hash* yang sesuai dan pengiriman *signature*. Bila kita lihat pada sistem *digital signature* (lihat Gambar 2.2 di atas), maka selain mengirimkan pesan, *signature*-nya juga juga perlu dikirimkan untuk keperluan verifikasi di penerima nantinya. Pada telepon hanya terdapat sebuah sambungan sehingga kita tidak dapat mengirimkan *signature* secara khusus. Untuk itu pada sinyal telepon *signature*-nya dimasukkan ke dalam sinyal pembicaraan sebagai *watermark*.

Pada *digital signature* misalnya untuk email, panjang pesan sudah tertentu dan pemberian *signature* dapat dilakukan setelah pesan selesai. Pada telepon, orang ingin dapat melakukan verifikasi secepat mungkin tanpa harus menunggu percakapan selesai. Untuk itu pada sinyal telepon, *signature* diberikan pada tiap bagian sinyal percakapan sehingga verifikasinya tidak harus menunggu sampai percakapan selesai.

Permasalahan lainnya dalam pemberian *signature* untuk sinyal telepon adalah penggunaan fungsi *Hash* yang tepat. Seperti yang telah disebutkan sebelumnya bahwa fungsi *Hash* standard seperti MD5 dan SHA-1 [6] tidak dapat digunakan untuk sinyal telepon karena sinyal telepon banyak mengalami gangguan seperti *noise* atau sambungan yang putus-putus.

3. Pemberian *Signature* Pada Sinyal Telepon

Pemberian *signature* pada sinyal telepon dibagi dalam proses *Hash* $H(m)$, enkripsi keluaran *Hash* $S_A(H(m))$, dan penyisipan *signature* sebagai *watermark*. Proses pemberian *signature* adalah menyisipkan keluaran fungsi *Hash* $H(m)$ yang telah dienkripsi $S_A(H(m))$ sebagai *watermark* dalam sinyal telepon. Gambar 3.1 menunjukkan proses yang terjadi pada pemberian *signature*.



Gambar 3.1 Pemberian *signature* pada sinyal telepon

Pada Gambar 3.1 diperlihatkan sebuah blok suara yang digunakan pada proses pemberian *signature*. Blok tersebut dibagi dalam dua bagian, pertama yaitu bagian yang akan dimasukkan ke dalam fungsi *Hash* dan bagian kedua yang akan disisipkan *signature*. Antara kedua bagian tersebut terdapat suatu selang waktu yang digunakan untuk membangkitkan *signature* yang akan disisipkan.

3.1 Fungsi **Hash** Untuk Sinyal Telepon

Beberapa persyaratan fungsi *Hash* untuk kriptografi adalah [7]:

1. untuk message m dan sebuah fungsi *Hash* H maka harus mudah untuk menghitung $H(m)$
2. untuk suatu h , maka sulit untuk menghitung m sehingga $h=H(m)$ (fungsi *Hash* tersebut harus satu arah)
3. untuk suatu m , sulit untuk mencari message lain m' sehingga $H(m')=H(m)$ (*collision free*)

Dari definisi di atas maka fungsi *Hash* yang digunakan dalam kriptografi harus sensitif dalam arti perubahan sedikit dari message m akan menghasilkan keluaran fungsi *Hash* yang benar-benar berbeda.

Untuk sinyal telepon, kita memerlukan fungsi *Hash* yang khusus karena seperti yang telah disebut sebelumnya bahwa sinyal telepon banyak mengalami gangguan seperti adanya *noise* atau sambungan yang terputus-putus. *Fridrich* dalam artikelnya *Robust Hash Function for Digital Watermarking* [7] membahas suatu fungsi *Hash* yang kuat (*robust*) untuk gambar digital. Keluaran fungsi *Hash* yang dihasilkan tetap sama walaupun gambar tersebut telah mengalami distorsi seperti kompresi, penskalaan atau operasi-operasi lainnya. Fungsi *Hash* semacam inilah yang cocok untuk dipakai pada sinyal telepon. Dalam tugas ini, fungsi *Hash* yang digunakan merupakan modifikasi dari fungsi *Hash* untuk gambar digital yang diusulkan oleh *Fridrich* [7].

Fungsi *Hash* yang dipakai untuk sinyal telepon bekerja dengan memproyeksikan sinyal suara ke beberapa pola sinyal *pseudo noise*. Dengan menggunakan sebuah kunci, *pseudo noise generator* (*PNG*) membangkitkan N buah pola sinyal *pseudo noise* $P^{(i)}$, $1 \leq i \leq N$. *Pseudo noise* tersebut adalah bilangan acak dengan distribusi *uniform* pada interval $[0,1]$. Selanjutnya dilakukan *low pass filtering* untuk mendapatkan sinyal yang *smooth* dan kemudian dihilangkan komponen DC-nya. Sinyal suara dan sinyal *pseudo noise* yang digunakan dalam perhitungan ini adalah sinyal diskrit sehingga sinyal telepon yang analog harus disampling terlebih dahulu.

Sebuah blok (B) diambil dari sinyal percakapan telepon. Dengan menganggap setiap blok dan sinyal *pseudo noise* tersebut sebagai vektor, blok sinyal suara (B) tersebut diproyeksikan ke setiap pola sinyal *pseudo noise* $P^{(i)}$, $1 \leq i \leq N$ dan nilai mutlaknya dibandingkan dengan nilai rata-rata proyeksi dari semua N pola sinyal *pseudo noise* untuk mendapatkan N bit *Hash* b_i :

$$\text{jika } |B \cdot P^{(i)}| < \text{rata-rata} \quad b_i = 0$$

$$\text{jika } |B \cdot P^{(i)}| \geq \text{rata-rata} \quad b_i = 1$$

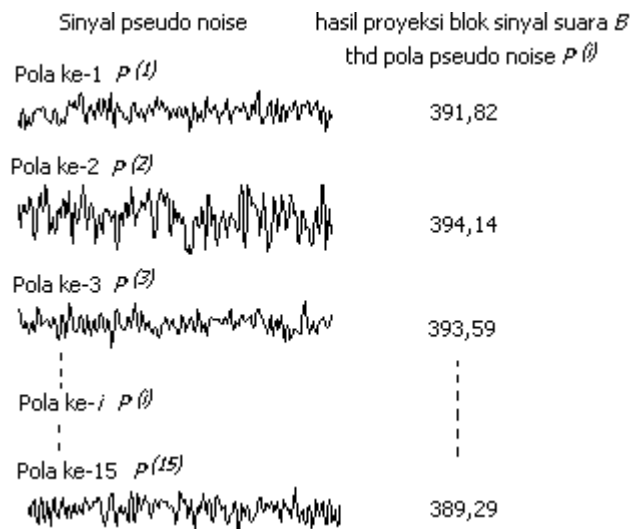
Nilai rata-rata dihitung setelah blok (B) diproyeksikan ke semua N pola sinyal *pseudo noise*. Nilai rata-rata ini dipakai untuk menentukan bit keluaran *Hash* b_i agar kira-kira setengah dari bit yang dihasilkan adalah 0 dan kira-kira setengahnya lagi adalah 1.

Hal ini dilakukan agar kita mendapatkan *highest information content* [7] dalam ekstraksi bit *Hash*. Gambar 3.2 di bawah ini menunjukkan contoh sinyal suara dan Gambar 3.3 adalah contoh dari beberapa pola sinyal *pseudo noise* dan hasil proyeksinya terhadap contoh sinyal suara pada Gambar 3.2.



Gambar 3.2 Contoh sinyal suara

Sumber: file *wjplug.wav* dari *Wendell Johnson's Voice: "Plogglies"* [8]



Gambar 3.3 Contoh pola sinyal *pseudo noise* yang digunakan dalam percobaan dan hasil proyeksinya terhadap sebuah blok sinyal suara

| Pola pseudo noise $P^{(i)}$ | Hasil Proyeksi dengan B | Keluaran bit Hash b_i |
|-----------------------------|---------------------------|-------------------------|
| 1 | 391,82 | 0 |
| 2 | 394,14 | 1 |
| 3 | 393,59 | 0 |
| 4 | 394,34 | 1 |
| 5 | 394,41 | 1 |
| 6 | 399,33 | 1 |
| 7 | 395,41 | 1 |
| 8 | 391,51 | 0 |
| 9 | 392,77 | 0 |
| 10 | 397,01 | 1 |
| 11 | 392,50 | 0 |
| 12 | 395,84 | 1 |
| 13 | 391,60 | 0 |
| 14 | 394,78 | 1 |
| 15 | 389,29 | 0 |

Tabel 3.1 Hasil proyeksi contoh blok sinyal suara dengan 15 pola *pseudo noise*

Sinyal *pseudo noise* pada Gambar 3.3 dibangkitkan oleh fungsi pembangkit bilangan acak dalam interval $[0,1]$ dengan distribusi *uniform* menggunakan *Matlab*.

Dari hasil proyeksi dengan 15 pola sinyal *pseudo noise* maka rata-rata nilai proyeksinya adalah 393,89. Setelah hasil proyeksi tersebut dibandingkan dengan nilai rata-ratanya maka didapatkan keluaran bit *Hash* b_i seperti pada Tabel 1 kolom ke-3. Keluaran fungsi Hash $H(m)$ merupakan barisan dari bit Hash b_i .

Operasi penskalaan pada sinyal suara tidak akan merubah keluaran dari fungsi *Hash* (bisa dilihat nanti dari hasil percobaan pada bagian 5.1) karena sinyal *pseudo noise* yang digunakan telah dihilangkan komponen DC-nya sehingga proyeksi terhadap sinyal *pseudo noise* hanya tergantung pada variasi dari sinyal suara.

Sinyal *pseudo-noise* yang digunakan dibangkitkan oleh sebuah kunci yang dalam hal ini digunakan waktu pembicaraan sebagai kunci. Hal ini dimaksudkan agar seseorang tidak dapat membuat rekaman palsu dengan cara menggabung-gabungkan potongan rekaman percakapan sebelumnya.

3.2 Enkripsi Keluaran Fungsi Hash

Keluaran dari fungsi *Hash* $H(m)$ kemudian dienkripsi menggunakan sistem kunci publik. Dalam percobaan ini digunakan suatu sistem kunci publik yang terkenal yaitu *RSA* [5] dengan panjang kunci sebesar 20 bit. Algoritma enkripsi asimetris lainnya tentu dapat digunakan untuk mengenkripsi keluaran fungsi *Hash* ini. Keluaran fungsi *Hash* $H(m)$ berupa urutan bit b_i 0 atau 1 dianggap sebagai bilangan biner dan diubah ke dalam nilai desimal sebelum dienkripsi. Contohnya keluaran fungsi *Hash* $H(m)$ pada Tabel 1 adalah 010111100101010 diubah ke dalam nilai desimal menjadi 12.074. Misalkan digunakan pasangan kunci rahasia S_A (67,68911) dan kunci publik P_A (1019,68911) maka:

$$S_A(H(m)) = 12.074^{67} \pmod{68.911} = 50.378$$

Kemudian kita ubah lagi ke dalam bilangan biner sepanjang 20 bit sesuai dengan panjang kunci yang digunakan sehingga didapat 00001100010011001010₂.

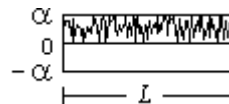
3.3 Penyisipan *Signature* Menggunakan Teknik *Watermarking*

Metoda *watermark* yang dipakai untuk menyisipkan *signature* harus tahan terhadap gangguan-gangguan yang mungkin dialami sinyal telepon seperti *noise* atau sinyal putus-putus dan juga tidak memerlukan sinyal asli dalam pendeteksian. Banyak teknik *watermarking* menggunakan metoda *spread spectrum modulation* dengan menggunakan sinyal *pseudo noise* untuk penyisipan dan pendeteksian [9] [10]. Ide yang dipakai adalah dengan menyebar *watermark* pada sebuah blok kemudian

memodulasinya dengan *signal pseudo-noise* [9] untuk kemudian disisipkan ke dalam sinyal yang akan diberi *watermark*. Jadi dalam sistem ini ada dua macam sinyal *pseudo noise* yang digunakan, yang pertama digunakan pada fungsi *Hash* dan yang kedua digunakan pada proses *watermarking*. Kedua macam sinyal *pseudo noise* tersebut sebenarnya sama saja yaitu berupa bilangan acak, namun mempunyai kegunaan yang berbeda.

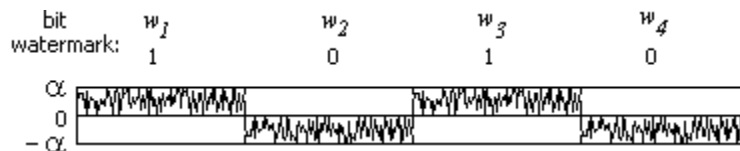
Sinyal *pseudo noise* untuk *watermark* dibangkitkan oleh kunci publik dari pembicara dengan menggunakan sebuah *pseudo noise generator*. Sinyal *pseudo noise* yang digunakan dalam percobaan ini sama dengan yang digunakan pada fungsi *Hash* yaitu berupa bilangan acak dalam interval $[0,1]$ dengan distribusi *uniform*.

Hasil keluaran fungsi *Hash* yang telah dienkripsi $S_A(H(m))$ adalah *signature* yang akan disisipkan ke dalam sinyal percakapan telepon sebagai *watermark*. Misalkan ada k buah bit *watermark* yang akan disisipkan maka kita perlu mempunyai k buah blok suara dengan panjang L untuk menyisipkan bit *watermark* tersebut dan sinyal *pseudo noise* dengan panjang L . Sinyal *pseudo noise* tersebut kemudian dikali dengan faktor penguatan α untuk menentukan amplituda sinyal *watermark* yang digunakan.



Gambar 3.4. Contoh sinyal *pseudo noise* untuk *watermark*

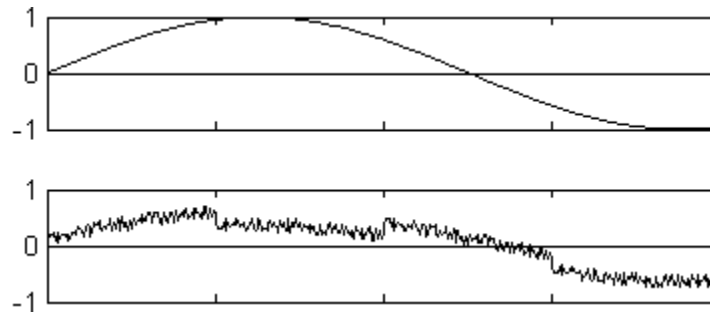
Untuk menyisipkan bit 1 kita kalikan sinyal *pseudo noise* dengan 1 dan untuk menyisipkan bit 0 kita kalikan sinyal *pseudo noise* dengan -1 . Misalkan kita ingin menyisipkan bit *watermark* $w_1 w_2 w_3 w_4 = 1010$ maka sinyal *watermark*-nya adalah :



Gambar 3.5 *Watermark* menggunakan sinyal *pseudo noise*

Dalam percobaan ini kita memiliki 20 bit *watermark* sesuai dengan panjang kunci yang digunakan untuk enkripsi. Setiap bit *watermark* tersebut akan disisipkan ke dalam blok suara dengan panjang $L = 0,125$ detik atau 1.000 sample (untuk frekuensi sampling 8.000 Hz).

Sinyal *pseudo noise* memiliki energi di semua komponen frekuensi sehingga kita dapat memilih bagian komponen frekuensi mana yang ingin disisipi *watermark*. Selanjutnya sinyal *watermark* tersebut ditambahkan ke dalam sinyal suara sehingga sinyal suara tersebut menjadi sinyal suara yang telah disisipi *watermark*. Penyisipan *watermark* diusahakan agar tidak mengganggu sinyal percakapan telepon. Dalam percobaan ini, sinyal *watermark* di-*high pass filter* kemudian disisipkan pada komponen frekuensi tinggi dari sinyal suara yang digunakan. Gambar 3.6 menunjukkan contoh sinyal suara asli dan yang sudah diberi *watermark* pada frekuensi tinggi.



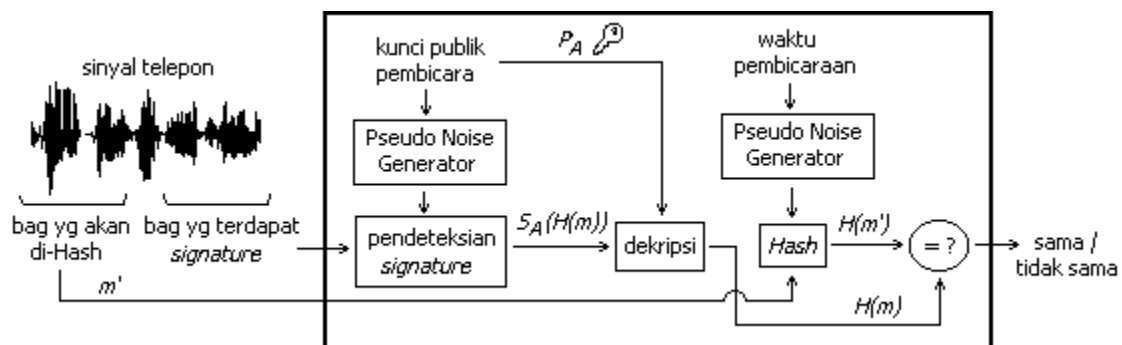
Gambar 3.6 Sinyal asli dan sinyal yang telah diberi *watermark*

4. Verifikasi *Signature*

Verifikasi dilakukan untuk mengetahui apakah sinyal yang diterima sesuai dengan *signature*-nya.

4.1 Verifikasi Pada Percakapan Telepon

Proses *verifikasi signature* pada sinyal telepon ditunjukkan oleh Gambar 4.1:



Gambar 4.1 Proses *verifikasi signature*

4.1.1 Proses Hash

Seperti pada proses pemberian *signature*, pada verifikasi *signature* sisi penerima juga dilakukan proses *Hash* pada sinyal yang diterima m' dan nanti hasil keluarannya $H(m')$ dibandingkan dengan *signature* yang disisipkan sebagai *watermark*. Proses *Hash* ini sama seperti pada pemberian *signature* dengan menggunakan waktu pembicaraan untuk membangkitkan N buah pola sinyal *pseudo noise* untuk diproyeksikan dengan sinyal suara.

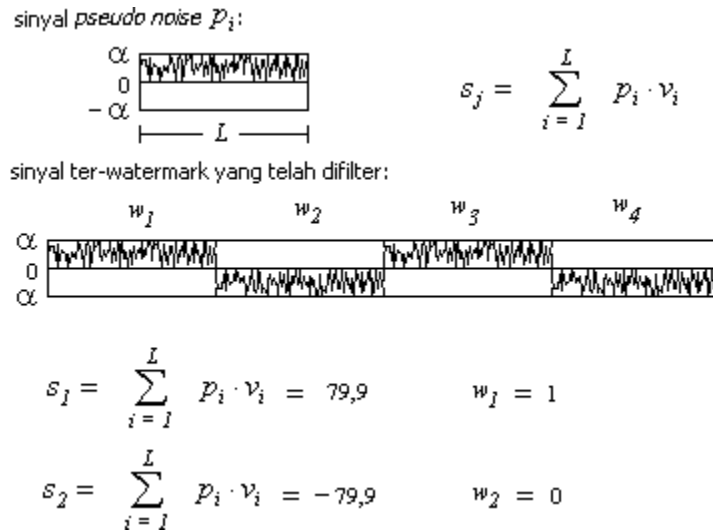
4.1.2 Pendeteksian *Signature*

Pendeteksian *signature* dilakukan dengan mem-*filter* sinyal suara yang telah diberi *watermark* untuk mendapatkan sinyal *watermark*. *Filtering* dilakukan pada frekuensi dimana *watermark* disisipkan. Dalam percobaan ini sinyal *watermark* disisipkan pada frekuensi tinggi (di atas sekitar 3.500 Hz) sehingga kita lakukan *high pass filtering* untuk mendapatkan sinyal *watermark*-nya kembali.

Dalam perhitungan ini, kita menggunakan sinyal diskrit sehingga sinyal telepon yang analog harus disampling terlebih dahulu. Misalkan p_i adalah sinyal *pseudo noise* yang digunakan untuk menyisipkan *watermark* dan v_i adalah blok sinyal yang telah di-*filter* untuk dideteksi *watermark*-nya dengan $1 \leq i \leq L$. Untuk setiap blok, kalikan setiap komponen p_i dengan v_i , kemudian dijumlah:

$$s_j = \sum_{i=1}^L p_i \cdot v_i$$

Bila s_j bernilai positif maka bit *watermark* yang terdeteksi adalah 1, sedangkan apabila s_j bernilai negatif maka bit *watermark* yang terdeteksi adalah 0. Gambar 4.2 di bawah ini menunjukkan salah satu contoh pendeteksian bit *watermark* w_j .

Gambar 4.2 Pedeteksi bit *watermark*

Metoda penyisipan dan pendeteksian *watermark* yang digunakan bisa saja diganti dengan metoda yang lain asalkan sesuai untuk sinyal telepon dan dalam pendeteksiannya tidak diperlukan sinyal aslinya.

4.1.3 Dekripsi Watermark

Hasil pendeteksian *signature* didekripsi menggunakan kunci publik. Sebagai contoh dalam percobaan ini, pendeteksian *signature* menghasilkan keluaran 00001100010011001010_2 lalu diubah ke dalam desimal menjadi 50.378 kemudian didekripsi dengan kunci publik pembicara P_A (1019,68911) sehingga dihasilkan:

$$H(m) = 50.378^{1019} \pmod{68.911} = 12.074 = 010111100101010_2$$

Ini merupakan keluaran fungsi *Hash* (atau bisa juga disebut *signature*) yang dikirim oleh pembicara A (terdiri atas 15 bit). Hasil ini selanjutnya dibandingkan dengan hasil dari keluaran fungsi *Hash* $H(m')$ yang dilakukan di sisi penerima. Bila $H(m) = H(m')$ maka berarti *signature* telah diverifikasi secara benar.

4.2 Verifikasi *Signature* Pada Potongan Rekaman

Pada potongan rekaman, kita mungkin tidak mengetahui bagian awal dari percakapan tersebut bila rekaman tersebut tidak dimulai dari awal pembicaraan sehingga kita tidak mengetahui dimana letak *signature* dan letak sinyal yang dipakai untuk

menghitung fungsi *Hash*. Untuk itu pertama kali kita harus mencari letak *signature* dalam potongan rekaman tersebut dengan cara mencari bagian sinyal yang mempunyai korelasi dengan sinyal *pseudo noise* yang digunakan untuk menyisipkan *watermark*.

Setelah letak *signature* diketahui maka kita juga dapat mengetahui letak sinyal yang digunakan untuk menghitung fungsi *Hash* (letaknya di sebelah kiri dari *signature*-nya). Pertama kita perlu memperkirakan waktu terjadinya pembicaraan untuk digunakan sebagai kunci dalam membangkitkan N buah pola sinyal *pseudo noise* untuk fungsi *Hash*. Selanjutnya verifikasi dilakukan menggunakan cara yang telah dijelaskan sebelumnya. Proyeksikan N buah sinyal *pseudo noise* tersebut dengan sinyal potongan rekaman. Bila keluaran *Hash* sama dengan *signature* (yang disisipkan menggunakan *watermark*) maka percakapan tersebut memang pernah dilakukan pada waktu tersebut. Bila kita mendapatkan waktu pembicaraan yang berurutan pada semua blok sinyal percakapan yang terdapat dalam rekaman maka dapat disimpulkan bahwa percakapan tersebut memang benar terjadi.

5. Hasil Percobaan

Percobaan dilakukan dengan menggunakan program *Matlab*, perhitungan dilakukan secara digital menggunakan komputer. Di sini tidak sampai dilakukan implementasi sesungguhnya pada telepon. Percobaan ini dilakukan untuk mengetahui batas ketahanan *signature* (ketahanan fungsi *Hash* dan *watermark*) terhadap operasi-operasi seperti penskalaan, *low pass filtering*, penambahan *noise*, dan sinyal yang putus-putus.

Untuk mensimulasikan sinyal percakapan telepon digunakan file wav (dapat dilihat pada Gambar 3.2) dengan frekuensi sampling 8.000 Hz dengan 8 bit tiap sampel. File ini diambil dari file *wjplog.wav* (sumber *Wendell Johnson's Voice: "Plogglies"*) [8].

5.1 Hasil Percobaan Untuk Fungsi *Hash*

Percobaan dilakukan pada fungsi *Hash* dengan panjang sebuah blok adalah 3 detik untuk sebuah *signature*. Hasil percobaan menunjukkan bahwa fungsi *Hash* yang digunakan tahan terhadap gangguan seperti:

- penskalaan: dari 0,1 sampai 100
- sinyal putus-putus: diuji dengan menghilangkan sampai sebanyak 5% dari sinyal suara (*random deletion*)
- *noise*: pemberian *noise* yang mengandung energi pada semua komponen frekuensi dengan amplituda sekitar 1% dari amplituda sinyal percakapan
- *low pass filtering*: dengan frekuensi *cut off* sampai 3.750 Hz menggunakan filter *Butterworth* orde 8.

5.2 Hasil Percobaan Untuk *Watermark*

Dalam percobaan, *watermark* disisipkan pada frekuensi tinggi yaitu di atas 3.500 Hz. Setiap bit *watermark* disebar pada sebuah blok yang panjangnya 0.125 detik atau 1.000 sampel untuk frekuensi sampling 8.000 Hz. Dari hasil percobaan didapatkan bahwa *watermark* tersebut tahan terhadap:

- penskalaan: dari 0,1 sampai 100
- *noise*: pemberian *noise* yang mengandung energi pada semua komponen frekuensi dengan amplituda sekitar 1% dari amplituda sinyal percakapan
- sinyal putus: diuji dengan menghilangkan sampai sebanyak 1% dari sinyal suara (*random deletion*)

Ketahanan *watermark* pada *filtering* tergantung pada proses penyisipan *watermark* karena dalam menyisipkan *watermark* kita dapat memilih untuk menyisipkan sinyal *pseudo noise* sebagai *watermark* di komponen frekuensi sesuai keinginan kita. Bila *filtering* tidak dilakukan pada komponen frekuensi yang digunakan oleh *pseudo noise* yang digunakan untuk menyisipkan *watermark*, maka *watermark* akan dapat terdeteksi. Apabila *filtering* mengenai komponen frekuensi yang digunakan oleh sinyal *pseudo noise* untuk menyisipkan *watermark* maka *watermark* tidak dapat terdeteksi.

6. Upaya Pemalsuan

Ada dua macam kasus pemalsuan, kasus pertama adalah seseorang menelepon secara langsung mengaku sebagai orang lain dan kasus kedua adalah membuat suatu rekaman pembicaraan palsu.

6.1 Cara Pemalsuan

Membuat pembicaraan dari potongan-potongan rekaman tidak dimungkinkan karena setiap pembicaraan telepon memiliki *signature* yang dibuat berdasarkan waktu pembicaraan berlangsung. Dengan menggabungkan potongan-potongan hasil rekaman maka akan dapat diketahui bahwa percakapan itu berasal dari potongan-potongan rekaman percakapan sebelumnya.

Seseorang yang memiliki kunci rahasia orang lain tentunya dapat memalsukan pembicaraan, namun hal ini sulit dilakukan karena di sini digunakan *RSA* untuk enkripsinya. Bila kunci yang digunakannya cukup panjang, maka akan diperlukan waktu yang lama untuk mencari kunci rahasianya.

Cara lain yang lebih mungkin dilakukan seorang pemalsu adalah dengan mengumpulkan pasangan keluaran fungsi *Hash* $H(m)$ dan keluaran fungsi *Hash* yang telah dienkrpsi $S_A(H(m))$ kemudian mengubah-ubah sinyal suara dalam batas-batas yang tidak terdengar agar keluaran fungsi *Hash*-nya sesuai dengan keluaran fungsi *Hash* $H(m)$ yang keluaran fungsi *Hash* yang telah dienkrpsi-nya $S_A(H(m))$ sudah diketahui. Hal mungkin dilakukan karena keluaran fungsi *Hash* yang digunakan tidak berubah secara drastis dengan adanya sedikit perubahan pada sinyal masukannya. Orang juga mungkin tidak akan menyadari bahwa telah terjadi sedikit perubahan dalam sebuah sinyal suara.

Pada rekaman, pemalsu mengganti sinyal rekaman asli dengan suara palsu kemudian melakukan perubahan-perubahan pada suara palsu tersebut dengan harapan dapat terjadi *collision* antara sinyal asli dengan suara yang dipalsunya sehingga dihasilkan keluaran fungsi *Hash* yang sama. Pada kasus dimana pemalsu menelpon seorang, maka dia perlu melakukan perubahan pada sinyal suaranya agar sama dengan salah satu pasangan keluaran *Hash* $H(m)$ dan yang telah dienkrpsinya $S_A(H(m))$. Hal ini dilakukan karena pemalsu tersebut tidak memiliki kunci rahasia dari orang yang

dipalsunya sehingga pemalsu tersebut tidak dapat mengenkripsi keluaran fungsi *Hash*-ya. Proses tersebut harus berjalan secara cepat dalam pembicaraan.

6.2 Keamanan Terhadap Upaya Pemalsuan

Seorang pemalsu berusaha mengubah-ubah suara (baik itu rekaman atau memang suara palsu yang di-*generate* sendiri) untuk menghasilkan keluaran fungsi Hash tertentu. Dia mungkin mengubah-ubah sinyal suara agar proyeksinya terhadap pola pertama dari sinyal pseudo *noise* menjadi berubah, namun setelah itu dia juga mungkin perlu melakukan perubahan lagi pada sinyal palsunya agar proyeksinya pada pola kedua dari sinyal pseudo *noise* juga berubah, perubahan pada sinyal suara yang kedua kali ini mungkin akan merubah hasil proyeksinya pada pola sinyal pseudo *noise* yang pertama. Bagaimana dengan hasil proyeksinya pada pola ketiga dan seterusnya dari beberapa pola sinyal *pseudo noise* yang digunakan dalam fungsi *Hash*. Pertanyaannya adalah seberapa banyak bit yang dapat diubah secara bersamaan (bukan satu persatu). Semakin banyak bit Hash yang digunakan, maka prosesnya akan semakin rumit. Bila hal ini mungkin dilakukan maka ini tentunya akan menurunkan kualitas sinyal palsu yang dihasilkan karena sinyal tersebut harus diubah-ubah agar untuk menghasilkan keluaran fungsi *Hash* tertentu.

Pada percobaan menggunakan komputer untuk fungsi *Hash* pada gambar digital, rata-rata sebanyak 13 bit (dari 50 bit *Hash*) dapat diubah secara bersamaan tanpa menyebabkan perubahan yang dapat dideteksi oleh mata berdasarkan model masking dari *Girod* [7].

7. Kesimpulan

Tugas ini telah membahas salah satu cara untuk melakukan verifikasi pada penelepon dengan cara menyisipkan *signature*. Dengan cara ini, penelepon dapat diverifikasi tanpa mengganggu percakapan. *Signature* yang digunakan berupa keluaran dari fungsi *Hash* yang dienkripsi menggunakan *RSA* kemudian disisipkan ke dalam sinyal telepon menggunakan teknik *watermarking* dengan menggunakan sinyal *pseudo noise*. Kuncinya adalah penggunaan fungsi *Hash* khusus untuk sinyal telepon yang tahan terhadap operasi penskalaan, *noise*, *filtering*, serta sinyal putus-putus. Hasil percobaan menunjukkan bahwa meskipun sinyal telepon telah mengalami gangguan, *signature* masih dapat dideteksi sehingga penelepon dapat diverifikasi.

Referensi

- [1] Bali Post. *Kambuh Lagi, Penipuan lewat Telepon di Praya*
<http://www.balipost.co.id/balipostcetak/2001/5/2/Nt5.htm> Mei 2001.
- [2] Mando Post Online. *Kasus Dokter Gadunga di Jakarta & Bandung*.
<http://www.mdopost.net/rangkuman-diskusi/jilid10/sekolah2/6667.html> April 2001
- [3] Suara Pembaruan Suara Pembaruan Daily. *Waspadai, Penelepon Catut Nama Pejabat Minta Transfer Uang*.
<http://www.suarapembaruan.com/News/2001/08/06/Nusantar/ns02/ns02.htm>
Agustus 2001
- [4] Howstuffworks. *How does Caller ID work?*
<http://www.howstuffworks.com/question409.htm>
- [5] Jan C. A. Van Der Lubbe. *Basic Methods of Cryptography*. Cambridge University Press. 1998.
- [6] R. Venkatesan, S. M. Koon, M. H. Jakubowski, and P. Moulin. *Robust Image Hashing*. Cryptography Group, Microsoft Research, 1 Microsoft way, Redmond, WA 98052-6399.
- [7] J. Fridrich and M. Goljan. *Robust Hash Function for Digital Watermarking*. Proceedings of the The International Conference on Information Technology: Coding and Computing (ITCC'00). IEEE. 2000.
- [8] Wendell Johnson. *Wendell Johnson's Voice: "Plogglies"*. N.J., Iowa City, July 18, 2000. <http://www.uiowa.edu/~cyberlaw/wj/wjaudio.html>
- [9] Frank Hartung and Bernd Girod. *Fast Public-Key Watermarking of Compressed Video*. Proceedings of the 1997 International Conference of Image Processing (ICIP'97). 1997.
- [10] I. Cox, J. Kilian, T. Leighton, and T. Shamoan. *Secure Spread Spectrum Watermarking for Multimedia*. Technical Report 95-10, NEC Research Institute, Princeton, NJ, USA, 1995.

Appendix : Source Code Matlab

Pemberian signature

```

% Penyisipan signature pada sinyal telepon
clear;

%---inisialisasi variabel-----
panjang_signature = 3; % panjang ucapan untuk sebuah signature (detik)
jml_bit_hash = 15; % jumlah keluaran bit Hash
panjang_kunci = 20; % untuk menentukan panjang watermark (tergantung panjang kunci)
waktu_pembicaraan = 0112311200; % waktu percakapan telepon (31 Des 2001 jam 12:00)
kunci_rahasia = 67; % kunci rahasia pembicara
kunci_publik = 1019; % kunci publik pembicara
n_perkalian_2bil_prima = 68911;

%-----Sinyal Telepon -----
%dalam percobaan ini kita gunakan suara rekaman
%frekuensi sampling 8000 kHz dengan 8 bit tiap sampel
[sinyal_telepon frek_sampling] = wavread('testfile');

%-----Fungsi Hash-----
% membangkitkan sinyal pseudo noise
temp(1:frek_sampling*jml_bit_hash*panjang_signature)=0;
rand('state', waktu_pembicaraan);
pn_hash = rand(size(temp));
clear temp;

% di-low pass filter dengan cut off 2000 Hz menggunakan Butterworth
[b,a] = butter(8,2000/4000);
temp = pn_hash;
pn_hash = filter(b,a,temp);
% dihilangkan komponen DC-nya
temp = fft(pn_hash);
temp(1)=0;
pn_hash=ifft(temp);
pn_hash = real(pn_hash); %hilangkan bagian imaginernya
pn_hash = transpose(pn_hash);

%proyeksi sinyal telepon terhadap pseudo noise
for i=1:jml_bit_hash,
    temp = pn_hash((i-1)*frek_sampling*panjang_signature+1:i*frek_sampling*panjang_signature);
    temp = sinyal_telepon(1:frek_sampling*panjang_signature).*temp;
    temp = abs(temp);
    bit_hash(i) = sum(temp);
end

%menetapkan treshold menggunakan nilai rata-rata
treshold = sum(bit_hash)/jml_bit_hash;

for i=1:jml_bit_hash,
    if bit_hash(i) < treshold
        bit_hash(i) = 0;
    else

```

```

        bit_hash(i) = 1;
    end
end

% output dari fungsi Hash
bit_hash

%----Enkripsi-----
%mengubah bilangan biner 15 digit ke decimal
desimal = 0;
for i=1:jml_bit_hash,
    if bit_hash(i)==1
        desimal = desimal + 2^(jml_bit_hash-i);
    end
end

% untuk enkripsi dengan kunci yang panjang, Matlab tidak mampu
% menangani bilangan yang terlalu besar

%enkripsi menggunakan kunci rahasia
chiper = mod(desimal^kunci_rahasia,n_perkalian_2bil_prima);

%mengubah dari desimal ke biner
for i=1:20,
    if chiper >= 2^(20-i)
        signature(i) = 1;
        chiper = chiper - 2^(20-i);
    else
        signature(i) = 0;
    end
end

%----Penyisipan signature dengan teknik watermark -----
clear temp;
temp(1:1000)=0;
rand('state',1019);
pn_wtm = rand(size(temp));
clear temp;

%sinyal pseudo-noisanya di high pass filter
[b,a] = butter(8,3500/4000, 'high');
temp=pn_wtm;
pn_wtm = filter(b, a, temp);

pn_wtm=transpose(pn_wtm);

%pilih faktor penguatan
alpha=0.8;

%itu sinyal suaranya di low pass dulu ya
[b,a] = butter(8,3000/4000);
vi = filter(b, a, sinyal_telepon((panjang_signature+1)*frek_sampling+1:(panjang_signature+1)*
frek_sampling+panjang_kunci*1000));
temp=vi;

% penyisipan watermark

```

```

for i=1:panjang_kunci,
    if signature(i) == 1
        vi((i-1)*1000+1:(i-1)*1000+1000) = temp((i-1)*1000+1:(i-1)*1000+1000)+alpha*pn_wtm*1;
    else
        vi((i-1)*1000+1:(i-1)*1000+1000) = temp((i-1)*1000+1:(i-1)*1000+1000)+alpha*pn_wtm*-1;
    end
end

sinyal_telepon((panjang_signature+1)*frek_sampling+1:(panjang_signature+1)*frek_sampling+
panjang_kunci*1000)=vi;

wavwrite(sinyal_telepon,frek_sampling,8,'sigatured_file');

```

Verifikasi Signature

```

clear;

%---inisialisasi variabel-----
panjang_signature = 3; % panjang ucapan untuk sebuah signature (detik)
jml_bit_hash = 15; % jumlah keluaran bit hash
panjang_kunci = 20; % untuk menentukan panjang watermark
waktu_pembicaraan = 0112311200; % waktu percakapan telepon (31 Des 2001 jam 12:00)
kunci_publik = 1019; % kunci publik pembicara
n_perkalian_2bil_prima = 68911;

%-----Sinyal Telepon -----
%dalam percobaan ini kita gunakan suara rekaman
%frekuensi sampling 8000 kHz dengan 8 bit tiap sampel
[sinyal_telepon frek_sampling] = wavread('sigatured_file');

%-----Fungsi Hash-----
% membangkitkan sinyal pseudo noise
temp(1:frek_sampling*jml_bit_hash*panjang_signature)=0;
rand('state', waktu_pembicaraan);
pn_hash = rand(size(temp));
clear temp;

% di low pass filter dengan cut off 2000 Hz menggunakan Butterworth
[b,a] = butter(8,2000/4000);
temp = pn_hash;
pn_hash = filter(b,a,temp);
% dihilangkan komponen DC-nya
temp = fft(pn_hash);
temp(1)=0;
pn_hash=ifft(temp);
pn_hash = real(pn_hash); %hilangkan bagian imaginernya
pn_hash = transpose(pn_hash);

%proyeksi sinyal telepon terhadap pseudo noise
for i=1:jml_bit_hash,
    temp = pn_hash((i-1)*frek_sampling*panjang_signature+1:i*frek_sampling*panjang_signature);
    temp = sinyal_telepon(1:frek_sampling*panjang_signature).*temp;
    temp = abs(temp);

```

```

    bit_hash(i) = sum(temp);
end

%menetapkan nilai treshold
treshold = sum(bit_hash)/jml_bit_hash;

for i=1:jml_bit_hash,
    if bit_hash(i) < treshold
        bit_hash(i) = 0;
    else
        bit_hash(i) = 1;
    end
end

% output dari fungsi hash
bit_hash

%-----Mendeteksi signature-----
vi_topi = sinyal_telepon((panjang_signature+1)* frek_sampling+1:(panjang_signature+1)*
frek_sampling+ panjang_kunci*1000);

%di high pass filter
clear temp;
[b,a] = butter(8,3500/4000,'high');
temp = vi_topi;
vi_topi = filter(b, a, temp);
vi_topi = transpose(vi_topi);

%generate sinyal pseudo noise sesuai dengan kunci pembicara
clear temp;
temp(1:1000)=0;
rand('state',1019);
pn_wtm = rand(size(temp));
clear temp;

% sinyal pseudo-noisanya di high pass filter dulu
[b,a] = butter(8,3500/4000, 'high');
temp=pn_wtm;
pn_wtm = filter(b, a, temp);
clear temp;

% pendeteksian watermark
for i=1:panjang_kunci,
    temp = pn_wtm.*vi_topi((i-1)*1000+1:(i-1)*1000+1000);
    if sum(temp)<0
        sj(i) = 1;
    else
        sj(i) = 0;
    end
end

% watermark hasil pendeteksian
sj

%-----Dekripsi pendeteksian watermark-----
% mengubah dari biner 20 digit ke decimal

```

```

desimal = 0;
for i=1:panjang_kunci,
    if sj(i)==1
        desimal = desimal + 2^(panjang_kunci-i)
    end
end

% untuk enkripsi dengan kunci yang panjang, Matlab tidak mampu
% menangani bilangan yang terlalu besar

%dekripsi
message = mod(desimal^kunci_public,n_perkalian_2bil_prima);

%mengubah dari desimal ke biner

for i=1:20,
    if message >= 2^(20-i)
        signature(i) = 1;
        message = message - 2^(20-i);
    else
        signature(i) = 0;
    end
end

%-----Verifikasi signature-----
% bila bit_hash sama dengan signature maka sinyal tersebut telah diverifikasi secara benar
bit_hash
signature

```