BAB II
TINJAUAN PUSTAKA

2.1 Steganografi

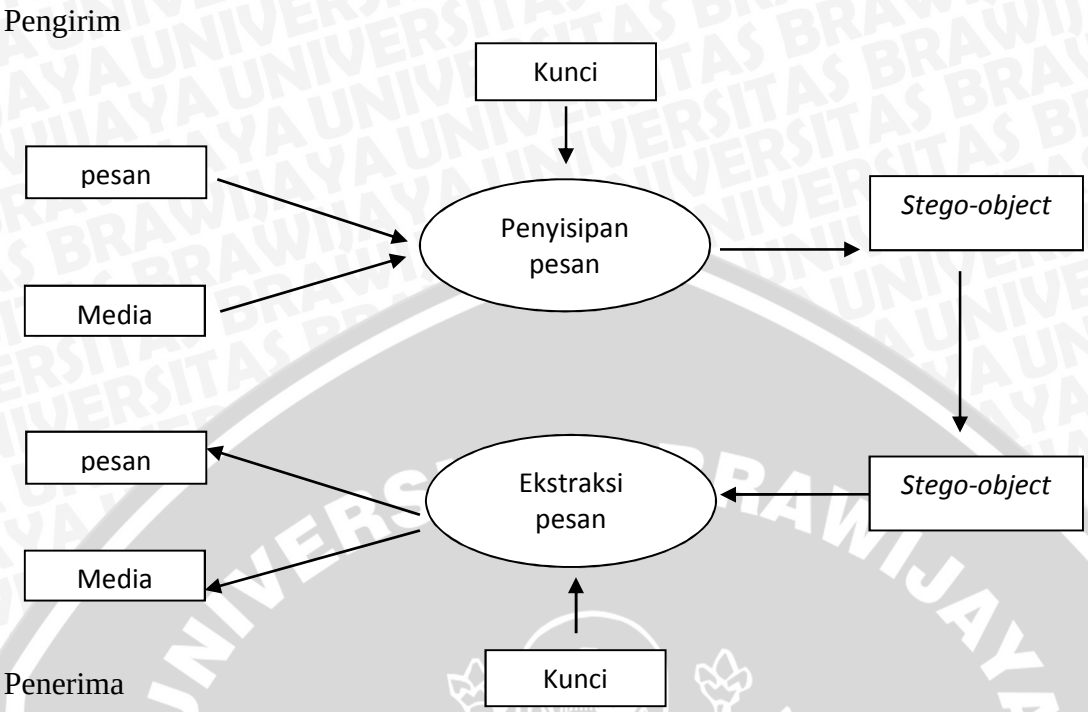
Steganografi secara umum adalah teknik menyisipkan pesan kedalam suatu media. Steganografi berasal dari bahasa yunani yaitu *stegos* yang berarti menyembunyikan dan *graptos* yang berarti tulisan sehingga steganografi berarti tulisan yang disembunyikan. Steganografi sudah dikenal oleh bangsa Yunani. Herodatus, penguasa Yunani, mengirim pesan rahasia dengan menggunakan kepala budak atau prajurit sebagai media. Dalam hal ini, rambut budak dibotaki, lalu pesan rahasia ditulis pada kulit kepala budak. Ketika rambut budak tumbuh, budak tersebut diutus untuk membawa pesan rahasia di balik rambutnya.

Bangsa Romawi mengenal steganografi dengan menggunakan tinta tak-tampak (*invisible ink*) untuk menuliskan pesan. Tinta tersebut dibuat dari campuran sari buah, susu, dan cuka. Jika tinta digunakan untuk menulis maka tulisannya tidak tampak. Tulisan di atas kertas dapat dibaca dengan cara memanaskan kertas tersebut. Saat ini di negara-negara yang melakukan penyensoran informasi, steganografi sering digunakan untuk menyembunyikan pesan-pesan melalui gambar (*images*), video, atau suara (audio).

Steganografi menggunakan sebuah berkas yang disebut dengan *cover*, tujuannya sebagai kamuflase dari pesan yang sebenarnya. Banyak format berkas *digital* yang dapat dijadikan media untuk menyembunyikan pesan. Pada jaman modern seperti saat ini, steganografi biasanya dilakukan dengan melibatkan berkas-berkas data *digital* seperti teks, audio, dan gambar.

Steganografi pada masa modern ini sering menggunakan media *digital* sehingga dikatan sebagai *digital staganography*. Media tersebut dapat berupa dokumen citra, audio, teks, maupun video. Terdapat beberapa istilah berkaitan dengan steganografi, yaitu :

1. Pesan yang disembunyikan (*hiddentext atau embedded message*)
2. Arsip untuk menyembunyikan pesan (*cover-object*)
3. Kunci rahasia (*stego-key*)
4. Arsip yang telah berisi pesan rahasia (*stego-object*)



Gambar 2.1 Diagram sistem steganografi.

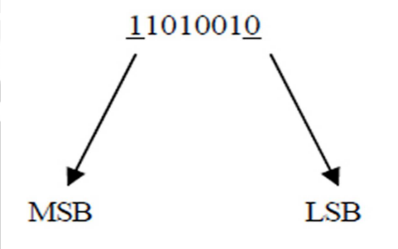
Pada gambar 2.1 pesan akan disisipkan ke dalam media tempat menyisipkan pesan (*cover-object*) oleh pengirim pesan dengan menggunakan suatu kunci, lalu media yang telah disisipi pesan (*stego-object*) akan diterima oleh penerima dan kemudian di ekstraksi menggunakan kunci yang sama sebagaimana yang digunakan ketika menyisipkan pesan.

2.1.1 Metode Steganografi

Dalam steganografi data dijital dikenal 2 metode steganografi yang digunakan. Pertama adalah metode yang bekerja pada ranah spasial, yaitu dengan memodifikasi langsung nilai *byte* dari *cover-object* dengan cara mensubtitusi. Metode ini dilakukan sedemikian rupa agar indra manusia tidak dapat mempersepsi media yang telah disisipi pesan. Salah satu metode dalam ranah spasial ini adalah metode *Least Significant Byte* (*LSB*). Kedua adalah metode yang bekerja dalam ranah frekuensi, yaitu dengan memodifikasi hasil transformasi sinyal dalam ranah frekuensi. Salah satu metode yang bekerja pada ranah frekuensi adalah teknik *spread spectrum.*

2.1.1.1 Least Significant Byte (LSB)

Least Significant Bit (LSB) merupakan salah satu metode steganografi yang paling sederhana, cepat dan mempunyai kapasitas penyisipan yang cukup besar. LSB menggunakan cara menyisipkannya pada bit rendah atau bit paling kanan (LSB) pada data *byte* yang menyusun berkas tersebut.



Gambar 2.2 MSB dan LSB.

Gambar 2.2 merupakan representasi biner 1 byte (8 bit) yang terdiri dari MSB (*Most Significant Bit*) terletak paling kiri dan LSB (*Least Significant Bit*) yang terletak paling kanan. Bit yang nantinya diganti adalah pada bagian yang paling kanan atau bagian LSB. Sebagai contoh, apabila akan disisipkan sebuah pesan teks kedalam sebuah segmen citra. Teks setelah di ubah menjadi bit adalah

0 1 0

Segmen citra setelah di ubah menjadi bit

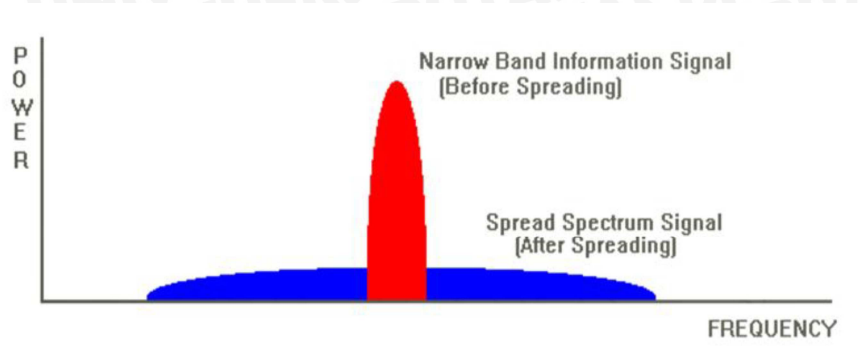
0 0 1 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 0 0 1 0

Maka setelah dilakukan proses penyisipan pesan pada citra dengan metode LSB menjadi

0 0 1 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 0 0 1 0

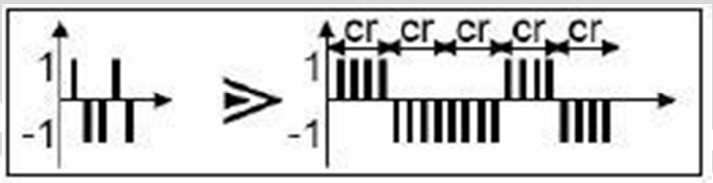
Metode LSB ini memanfaatkan kelemahan indera manusia yang tidak dapat membedakan sedikit perubahan yang terjadi. Dalam perkembangannya, proses penyisipan pesan diperkuat dengan cara membangkitkan bilangan acak. Bit-bit pesan tidak digunakan mengganti *byte* yang berurutan namun dipilih susunan *byte* yang acak. Misalnya terdapat 100 *byte* dan 7 bit data yang akan disembunyikan maka *byte* yang akan diganti pada media penyisipan bit LSB nya dipilih secara acak misalkan *byte* 47, 53, 15, 7, 69, 32, 87.

2.1.1.2 *Spread Spectrum*



Gambar 2.3 Sebaran Spektrum Audio pada *Spread Spectrum.*

Spread spectrum adalah sebuah teknik transmisi dimana kode *psedo noise* digunakan sebagai gelombang modulasi untuk "menyebarkan" energi sinyal melalui sebuah *bandwith* yang jauh lebih besar daripada *bandwith* sinyal informasi. Pada awalnya metode ini dikembangkan untuk kepentingan militer dan intelejen. Ide dasarnya adalah untuk menyebarkan sinyal informasi melalui *bandwidth* yang lebih luas untuk mencegah dilakukannya pencegatan informasi dan gangguan-gangguan lainnya. Istilah *spread spectrum* digunakan karena pada sistem ini sinyal yang ditransmisikan memiliki *bandwidth* yang jauh lebih lebar dari *bandwidth* sinyal informasi. Proses penebaran *bandwidth* sinyal informasi ini disebut *spreading*. Penyebaran ini berguna untuk menambah tingkat redudansi. Besaran redudansi ditentukan oleh faktor pengali cr yang bernilai skalar. Panjang bit-bit hasil penyebaran ini menjadi cr kali panjang bit-bit awal, seperti ilustrasi pada Gambar 2.4 .



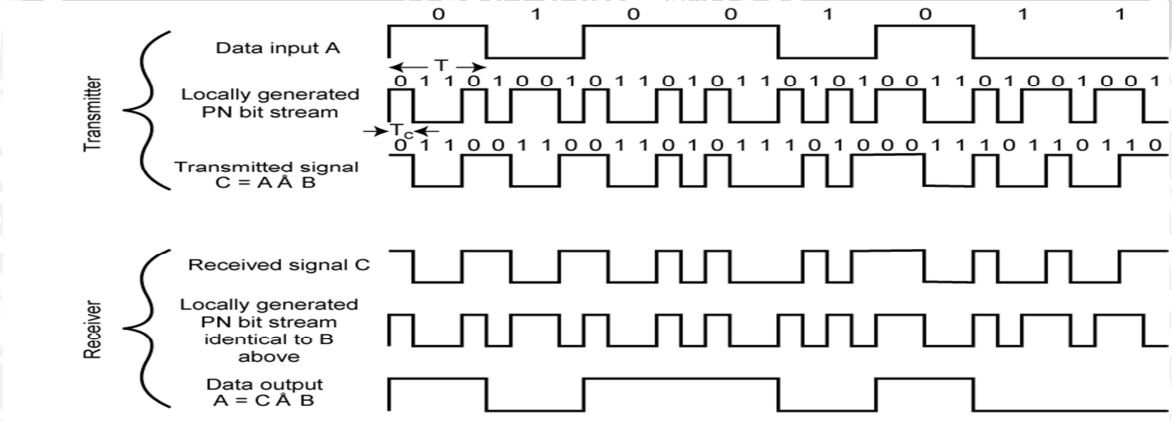
Gambar 2.4 Penyebaran Bit-bit dengan Faktor Pengali cr.

2.1.1.2.1 *Direct Sequence Spread spectrum* (DSSS)

Direct Sequence Spread Spectrum, setiap bit dalam sinyal asli direpresentasikan dengan banyak bit dalam sinyal yang ditransmisikan menggunakan sebuah kode penyebaran (cr). *Direct Sequence Spread Spectrum* dipilih karena adanya kemudahan dalam mengacak data yang akan di-*spreading*. Dalam DSSS *spreading* hanya menggunakan sebuah generator *noise* yang periodik yang di sebut *Pseudo Noise Generator*. Kode yang digunakan pada sistem *spread spectrum* memiliki sifat acak tetapi periodik sehingga disebut sinyal acak semu (*pseudo random*). Kode tersebut bersifat sebagai *noise* tapi deterministik sehingga disebut juga *noise* semu (pseudo *noise*). Pembangkit sinyal kode ini disebut Pseudo Random Generator (PRG) atau pseudo *noise* generator (PNG). Dalam skema ini, masing masing bit pada sinyal yang asli ditampilkan oleh bit- bit multipel pada sinyal yang ditransmisikan dengan menggunakan kode penyebaran (cr). Kode penyebaran akan menyebarkan sinyal sepanjang band frekuensi yang lebih luas. Oleh karena itu, kode penyebaran 10-bit menyebarkan sinyal sepanjang band frekuensi yang 10 kali lebih besar dibandingkan kode penyebaran 1-bit.

Proses penyebaran ditulis dalam persamaan 2.1 :

$$\boldsymbol{B = \{bi \mid bi = aj, j \cdot cr \leq i < (j+1) \cdot cr\}} \qquad \textbf{(2.1)}$$



Gambar 2.5 Contoh *Direct Sequence Spread Spectrum.*

Dalam Gambar 2.5 diperlihatkan *Direct Sequence Spread Spectrum* sinyal informasi yang telah disebar akan dimodulasikan dengan sinyal acak semu (*pseudo random)*

dengan menggunakan cara *exclusive*-OR (XOR). XOR mengikuti aturan sebagai berikut.

$0 \oplus 0 = 0$ $0 \oplus 1 = 1$ $1 \oplus 1 = 1$ $1 \oplus 0 = 0$ (2.2)

Proses modulasi ditulis dalam persamaan 2.3 :

$$wi = \alpha \cdot bi \cdot pi \quad (2.3)$$

Dengan,

wi : pesan termodulasi

α : faktor penguat

bi : pesan yang sudah di-*spreading*

pi : PN sequence

2.2 Audio WAV

WAV disebut dengan sebutan singkat untuk *Waveform Audio Format.* standar format berkas audio yang dikembangkan oleh Microsoft dan IBM. WAV merupakan varian dari format bitstream RIFF dan mirip dengan format IFF dan AIFF yang digunakan komputer Amiga dan Macintosh.

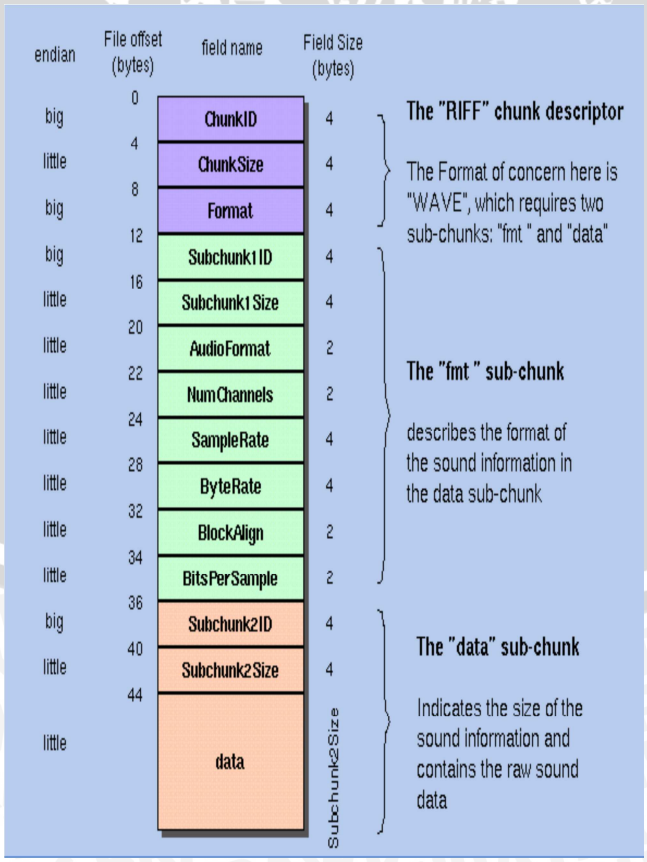
Baik WAV maupun AIFF kompatibel dengan operating system Windows dan Macintosh. meski WAV dapat menampung audio dalam bentuk terkompresi, umumnya format WAV merupakan audio yang tidak terkompresi. Sehingga jika ingin menyimpan dan dapat terbaca oleh sebuah komputer maka suara tersebut harus disimpan dalam bentuk *digital* hal ini bisa dilakukan dengan mengambil sampel sejumlah bagian gelombang per detiknya, lalu disimpan ke computer dalam bentuk format WAV .

Dalam Gambar 2.6 ditunjukan garis merah menandakan sampel yang diambil pada gelombang. Dapat dilihat bahwa semakin banyak sampel yang diambil tiap detiknya, sampel akan semakin mendekati bentuk gelombang aslinya, artinya kualitas akan semakin baik.

Gambar 2.6 Grafik Sampling Gelombang.

Jenis format Wave ini merupakan jenis *file Wave* yang paling umum dan hampir dikenal oleh setiap program. Format *Wave* PCM (*Pulse Code Modulation*) adalah *file wave* yang tidak terkompresi, akibatnya ukuran berkas sangat besar jika berkas audio mempunyai durasi yang panjang.

Pada audio berformat WAV ini terdiri atas 2 buah SubChunk2 yaitu: “fmt ” dan “data”. Dalam hal ini SubChunk “fmt “ menggambarkan format *data sound* sedangkan SubChunk “data” terdiri atas ukuran besar data dan *data sound* sebenarnya. Berikut ini diagram Gambar 2.7 yang menggambarkan format *file Wave* PCM.



Gambar 2.7 Format *file* WAV.

2.3 Pembangkitan Bilangan *Pseudo random*

Bilangan acak (*random*) banyak digunakan dalam kriptografi. Pembangkitan bilangan acak pada steganografi, dapat digunakan untuk menentukan kunci penyisipan dan ekstraksi data dari berkas media. Namun, sangat sulit untuk menghasilkan bilangan yang benar-benar acak dengan menggunakan komputer. Komputer hanya mampu menghasilkan bilangan semu acak (*pseudo random*). Deret bilangan *pseudo random* adalah deret bilangan yang kelihatan acak dengan kemungkinan pengulangan yang sangat kecil atau periode
pengulangan yang sangat besar.

2.3.1 *Linier Congretial Generator*

Linear Congruental Generator (*LCG*) salah satu algoritma pembangkitan bilangan pseudorandom. Algoritma ini merupakan pembangkit bilangan acak yang sederhana yang diciptakan oleh D. H.Lehmer pada tahun 1951. Deret bilangan bulat dalam *LCG* diformulasikan sebagai berikut :

$$X_n = (aX_{n-1} + b) \bmod m \qquad (2.4)$$

Dengan :

X_n = bilangan acak ke-n dari deretnya
X_{n-1} = bilangan acak sebelumnya
a = faktor pengali
b = *increment*
m = modulus

Untuk memulai bilangan acak ini dibutuhkan sebuah bilangan bulat X_0, yang dijadikan sebagai nilai awal (bibit pembangkitan). Bilangan acak pertama yang dihasilkan selanjutnya menjadi bibit pembangkitan bilangan bulat acak selanjutnya. Jumlah bilangan acak yang tidak sama satu sama lain (unik) adalah sebanyak *m*. Semakin besar nilai m, semakin kecil kemungkinan akan dihasilkan nilai yang sama.
Dengan demikian nilai X_0 terdefinisi pada :

$0 = X_0 = n\text{-}1$, $n = 1, 2, 3, \ldots$

2.4 *Fast Fourier Transform*

Pada tahun 1960, J. W. Cooley dan J. W. Tukey, berhasil merumuskan suatu teknik perhitungan algoritma Fourier Transform yang efisien. Teknik perhitungan algoritma ini dikenal dengan sebutan *Fast Fourier Transform* atau lebih populer dengan istilah FFT yang diperkenalkan oleh J.S.Bendat dan A.G.Piersol pada 1986. *Fast Fourier Transform* adalah Transformasi Fourier Cepat yang merupakan sumber dari suatu algoritma untuk menghitung *Discrete Fourier Transform* (transformasi *fourier diskrit* atau DFT) dengan cepat, efisien dan inversnya.

DFT merupakan metode transformasi matematis untuk sinyal waktu diskrit ke dalam domain frekuensi. Secara sederhana dapat dikatakan bahwa DFT merupakan metode transformasi matematis sinyal waktu diskrit, sementara FFT adalah algoritma yang digunakan untuk melakukan transformasi tersebut.

Secara matematis, DFT dapat dirumuskan sebagai berikut :

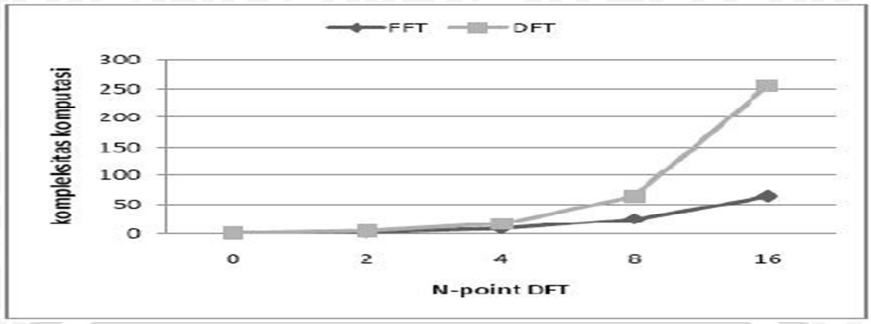
$$X[k] = \sum_{n=0}^{N-1} x[n].W_N^{nk} \quad ; k = 0,1,2,\dots,N-1 \tag{2.5}$$

Sementara itu, transformasi balikan atau *Inverse Discrete Fourier Transform* (IDFT) dapat dirumuskan sebagai berikut :

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k].W_N^{-nk} \quad ; n = 0,1,2,\dots,N-1 \tag{2.6}$$

FFT dipergunakan untuk mengurangi kompleksitas transformasi yang dilakukan dengan DFT. Sebagai perbandingan, bila kita menggunakan DFT, maka kompleksitas transformasi kita adalah sebesar $O(N^2)$, sementara dengan menggunakan FFT, selain waktu transformasi yang lebih cepat, kompleksitas transformasi pun menurun, menjadi O(N log (N)). Untuk jumlah *sample* yang sedikit mungkin perbedaan kompleksitastidak begitu terasa, namun lain ceritanya bila kita mengambil jumlah *sample* yang sedikit lebih banyak. Misalnya kita hanya mengambil 2 *sample*, dengan menggunakan DFT, tingkat kompleksitas transformasi kita adalah 4, sementara dengan menggunakan FFT kompleksitasnya sebesar 0,602. Perbedaan yang semakin mencolok tampak bila kita mengambil jumlah *sample* yang lebih banyak lagi, misalnya kita ingin meninjau 64 titik *sample*, maka kompleksitas dengan menggunakan DFT adalah sebesar 4096, sementara dengan menggunakan FFT kompleksitasnya menjadi 115,6. Perbedaan yang sangat

besar, melihat perbandingan yang mencapai hampir 40 kali lipatnya. Kompleksitas transformasi ini terutama menjadi vital saat diimplementasikan pada perangkat riil. Perbandingan kompleksitas DFT dan FFT dapat dilihat pada Gambar 2.8

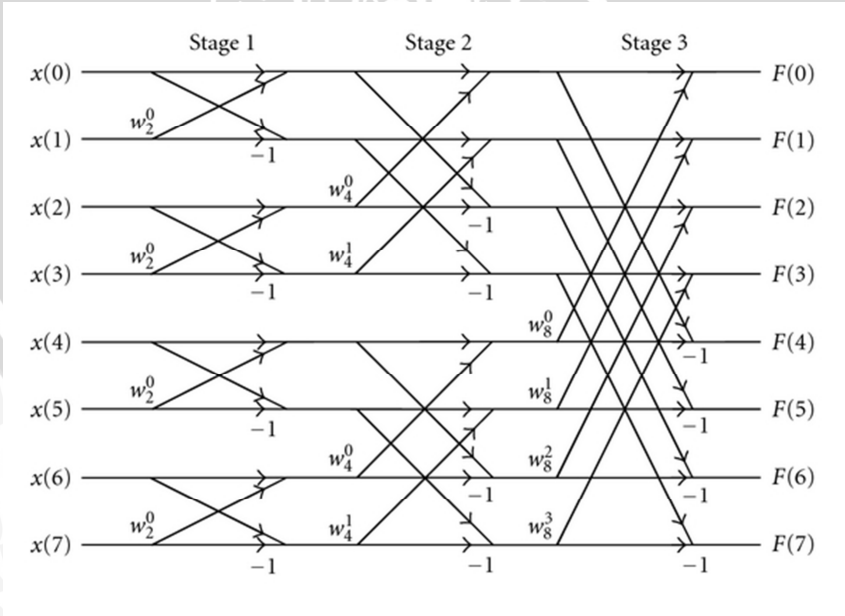


Gambar 2.8 Perbandingan kompleksitas DFT dan FFT.

2.4.1 *Fast Fourier Transform radix* 2

Dalam transformasi sinyal waktu diskrit dengan menggunakan FFT dikenal istilah *butterfly*. *Butterfly* adalah elemen terkecil dari suatu operasi FFT. Masing-masing jenis pendekatan algoritma FFT dapat dikelompokkan lagi menjadi beberapa jenis. Beberapa jenis yang sering digunakan dalam pendekatan FFT salah satu nya adalah Radix-2.

Radix-2 berarti, sejumlah N-*sample* yang akan ditransformasikan dibagi menjadi 2 kelompok dalam tiap kali rekursi yang terjadi. Pada gambar berikut ini dapat dilihat bagaimana proses kerja Radix-2



Gambar 2.9 FFT Radix 2.

Dari contoh di atas kita dapat melihat, untuk mentransformasikan 8 titik *sample* dengan menggunakan Radix-2, maka dibutuhkan 3 tahap proses, angka ini diperoleh dari $\log_2 8 = 3$