Steganalysis of Hydan

IFIP Sec 2009 18-20 May, Paphos Cyprus

Jorge Blasco[1], Julio C. Hernández-Castro[1], Juan M.E. Tapiador[1], Arturo Ribagorda[1] and Miguel A. Orellana-Quirós[2]

[1]Carlos III University of Madrid, Computer Science Department
[2]Ministry of Economy, Madrid

May 18th, 2009

Outline

Steganography

- Art and Science
- Hide the existence of messages
- Described by Simmons in the Prisoners problem [12]
- Desired properties of steganographic systems
  - High Capacity
  - Security Against
    - Passive attacks (detection)
    - Active attacks (modification or deletion)
- Applications
  - Covert Communications
    - Spies
    - Dissidents
    - Malicious employees
  - Add information to legacy formats
  - ...

Steganalysis

- Studies security of stego-systems
- Focuses mainly on detection (Passive attacks)
- Types
  - Blind: detects any algorithm
  - Targeted: detects one algorithm
- Methods
  - Find weakness on the algorithm (As in cryptography)
  - Statistical Analysis

Outline

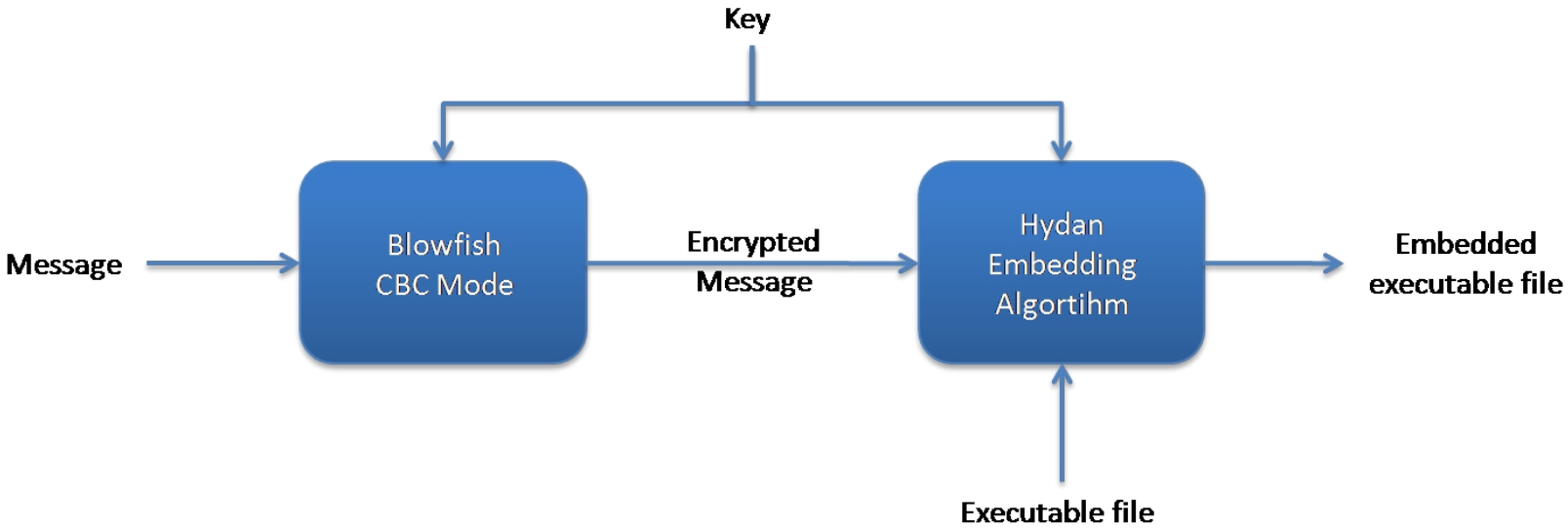
Information Hiding in Executable Files

- Steganography
  - Hydan [2]
  - Stilo [1]
- Steganalysis
  - Only described for Stilo
  - Code Transformation Signatures
- Watermarking [13]
  - Copy detection
  - Integrity check

Outline

Basics of Hydan

- Hides information in executable files
- Does not change size nor functionality
- It is able to hide 1 bit per 110 bits of executable file
- Uses redundancy on instructions



Functionality-Equivalent Instruction Sets I

Definition

A set of *functionality-equivalent instructions* is a group of instructions in which any can be replaced for other without loss of functionality

- Equivalent instructions codify different values

Example

Encodes 0	Encodes 1
and r2, 0x22	*and r2, 0x22*
xor r1, 0x34	*xor r1, 0x34*
add r1, 0x2A	*sub r1, -0x2A*

Functionality-Equivalent Instruction Sets II

- Hydan uses 31 sets of functionality-equivalent instructions
- Replacements are more complex than our example
- Instruction replacement changes distribution of instructions on executable files

Group	Instructions	Group	Instructions
toac8	5	*toac32*	5
rrcmp32	2	*toasxc8*	7
addsub8	2	*addsub8-2*	2
addsub32-2	2	*addsub32-3*	2
xorsub32	4	*add8*	2
adc8	2	*adc32*	2

Table: Examples of equivalent instructions sets used in *Hydan*

Outline

Overview

- Goal: Tell apart clean files from stego-files
- How: Model the instruction selection made by compilers
  - Defining reference distributions for each instruction set
- Procedure
  - Analyze 1261 clean executable files from an Ubuntu distribution
  - Calculate average distribution for each equivalent instruction set
  - Use a χ^2 statistic to measure distribution difference

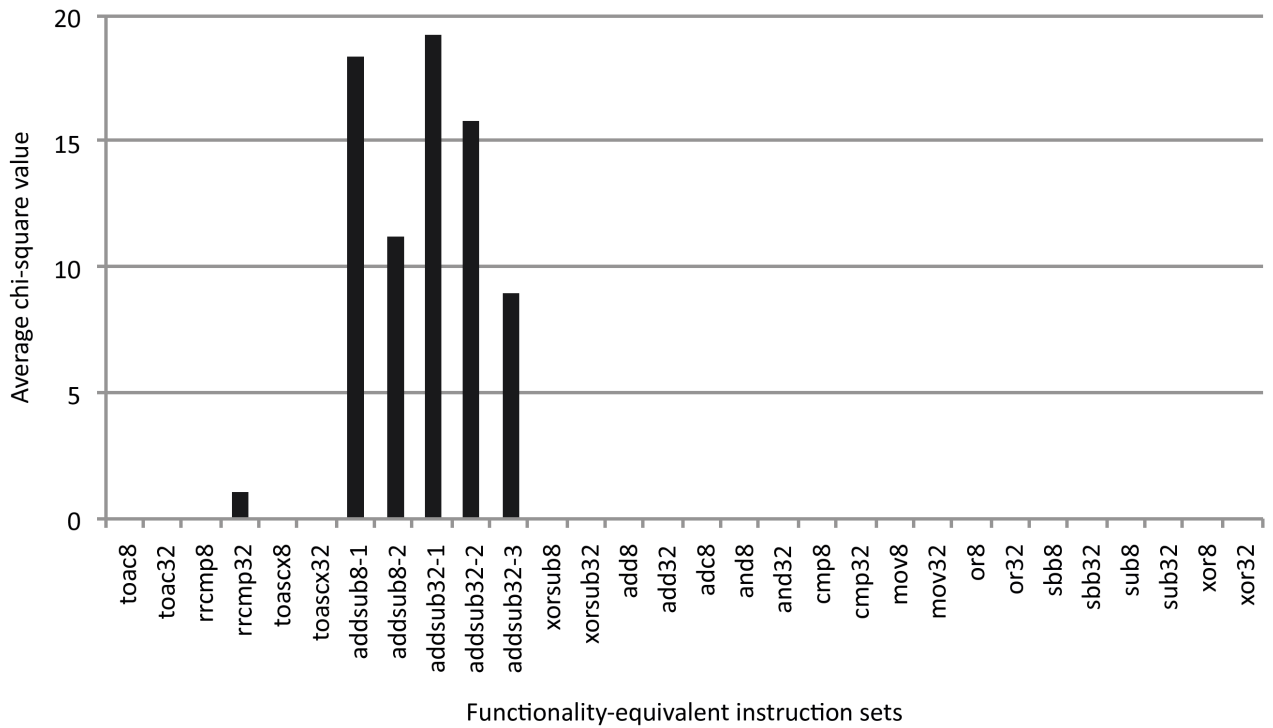
Distribution of instructions inside instruction set

Average distribution of instructions on the *toac32* set

Instruction	Frequency
test r/m32, r32	100.0%
or r/m32, r32	0.0%
or r32, r/m32	0.0%
and r/m32, r32	0.0%
and r32, r/m32	0.0%

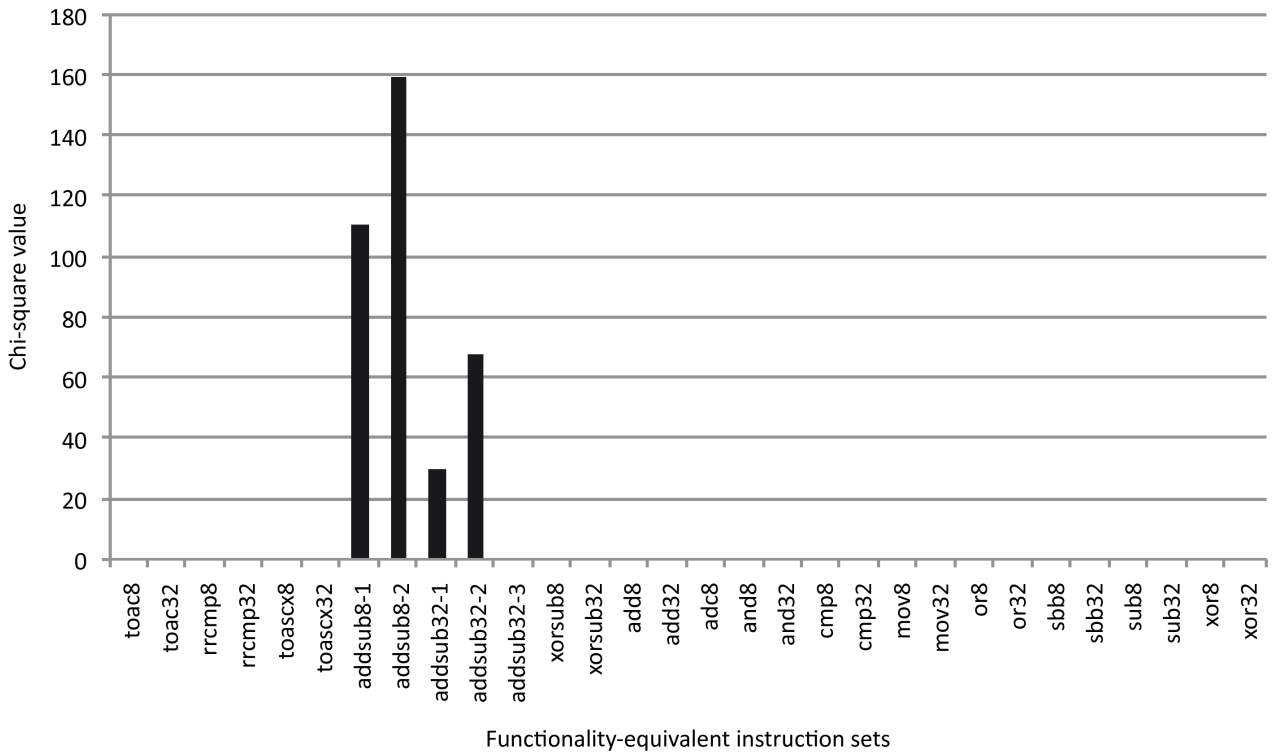
Statistical Analysis Result: Average

Average χ^2 statistic for each of the equivalent instructions sets for the selected files compiled with *gcc* for a *x86* processor



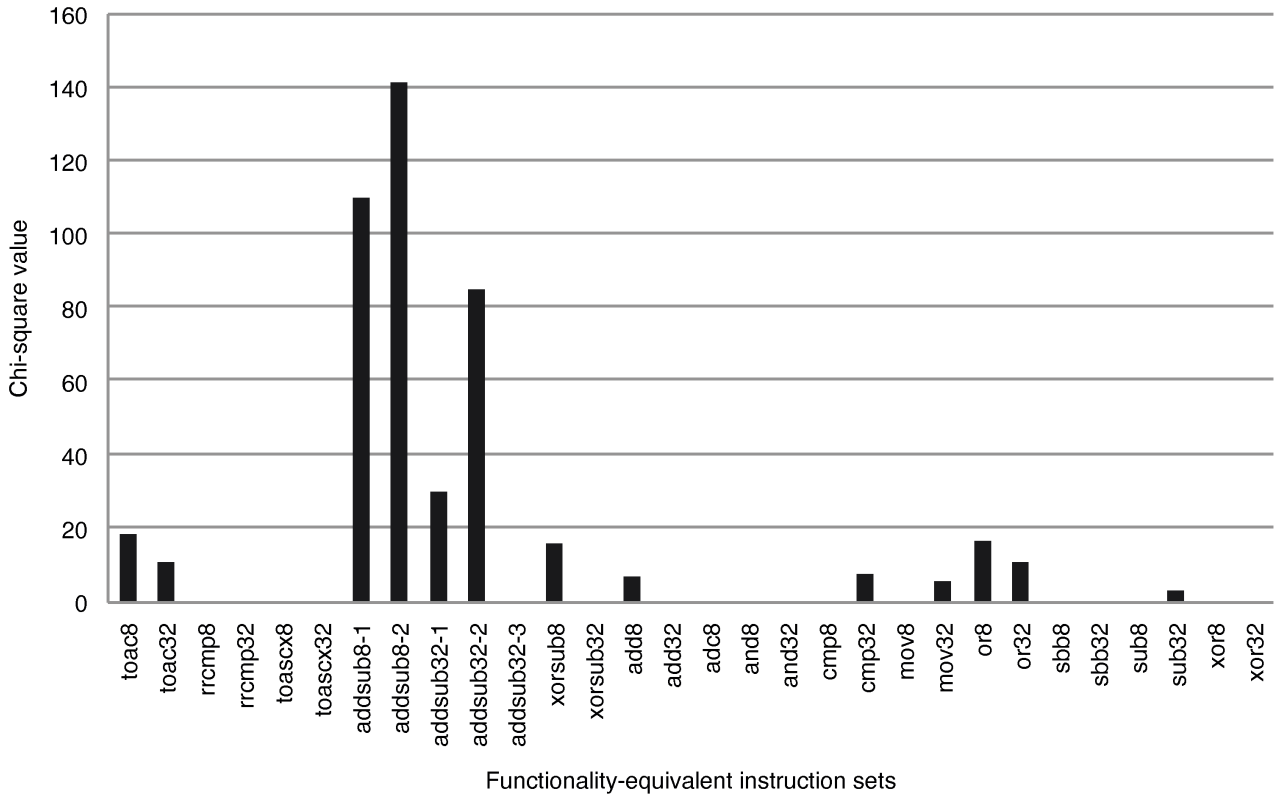
Statistical Analysis Result: An example I

χ^2 statistics for each of the equivalent instructions sets in *apt-get*



Statistical Analysis Result: An example II

χ^2 statistics for each of the equivalent instructions sets in *apt-get* with embedded information

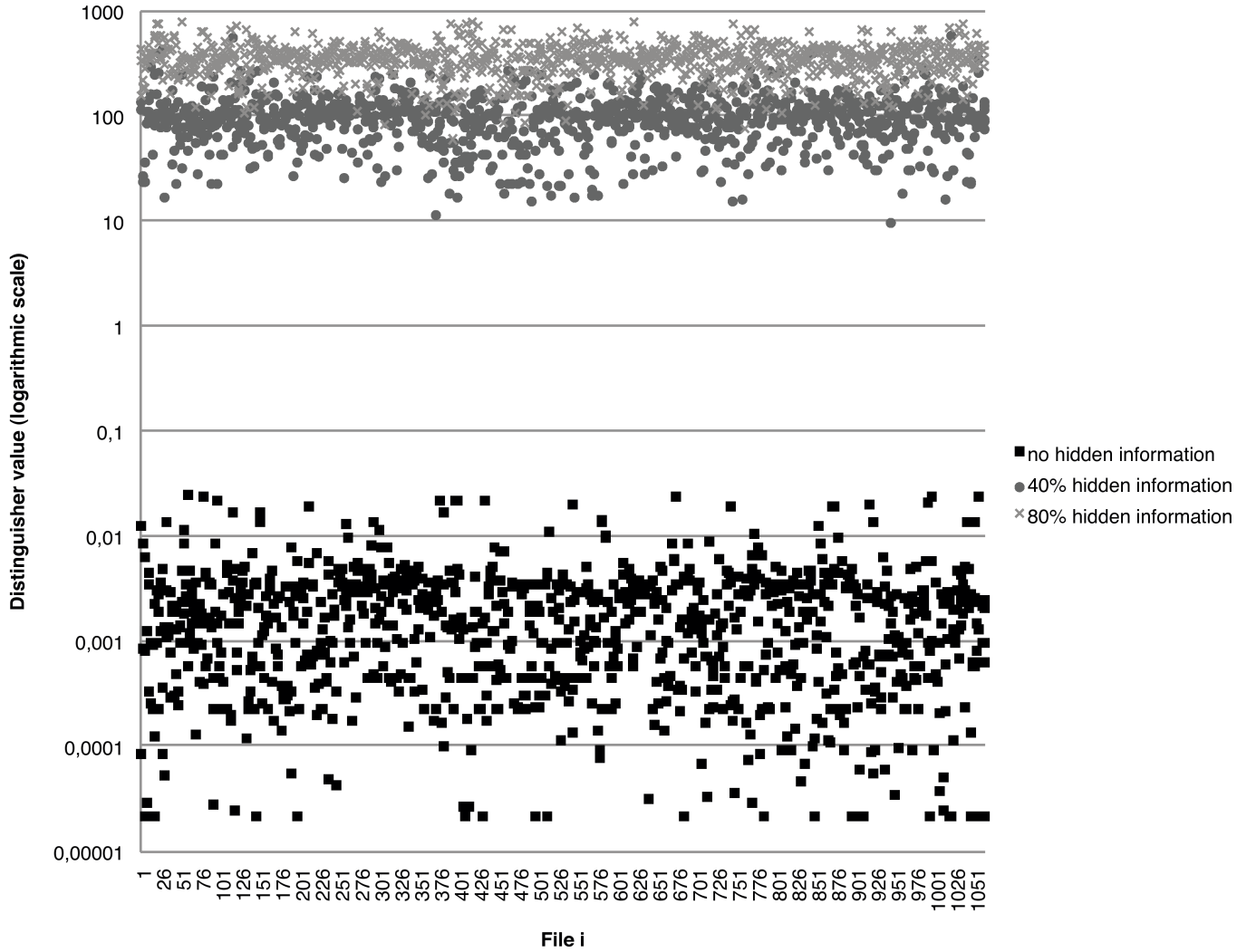


Distinguisher Design

- Use a χ^2 statistic to measure anomalies in instruction ocurrence
- Select only instruction sets which distribution is *constant* across files
  - 8 instruction sets discarded
- Distinguisher value
  - $D(file) = \sum_{i=0}^{n} \chi^2_{instruction\ set_i}$
- Threshold uses mean and standard deviation
  - $T = Mean_{cl} + S.Dev._{cl} = 0.000604 + 0.024571 = 0.025175$
- File is marked if distinguisher value is bigger than threshold

Outline

Distinguisher results



Outline

Conclusions

- Our system successfully distinguishes executable files with embedded information
- By using distribution of equivalent instruction sets
- In our experimental setup, it detects **all the files** with hidden information
- To have even a slight probability of passing unnoticed
  - Capacity should be limited to less than 35%
  - Makes *Hydan* only suitable for watermarking
  - A new statistical analysis may detect them
- Distinquisher depends on compiler, target OS and target platform, but is always doable with our approach.

Future Work

- Extend our approach to other compilres, OS and platforms
- Use and improve our techniques against Stilo
- Move towards blind steganalysis techniques
- Design and analyze new algorithms for information hidding in executable files

Outline


Anckaert B., De Sutter B., Chanet D., De Bosschere K.: Steganography for Executables and Code Transformation Signatures.
Lecture Notes in Computer Science **3506**, 425–439 (2005)


El-Khalil, R.: Hydan: Hiding Information in Program Binaries (2003).
Lecture Notes in Computer Science **3269**, 187–199 (2004)
`http://crazyboy.com/hydan/.Cited20Oct2008`


Hernandez-Castro J.C., Lopez I.B., Tapiador J.M.E., Ribagorda A.: Steganography in Games.
Computers and Security **25**(1), 64–71 (2006)


Johnson N.F., Jajodia S.: Exploring steganography: Seeing the unseen.
Computer **31**(2), 26–34 (1998).


Kipper, G.: Investigator's Guide to Steganography.
CRC Press (2004)


Murdoch S.J., Lewis S.: Embedding Covert Channels into TCP/IP.
Lecture Notes in Computer Science **3727**, 247–261 (2005)


Naor M., Yung M.: Universal One-Way Hash Functions and Their Cryptographic Applications.
Proceedings of the twenty-first annual ACM symposium on Theory of computing, pp. 33–43. ACM, New York, NY, USA (1989).

 Peterson W., Brown D.: Cyclic Codes for Error Detection.
Proceedings of the IRE **49**(1), 228–235 (1961)

 Petitcolas F.A.P., Anderson R.J., Kuhn M.G.: Information Hiding:A Survey.
Proceedings of the IEEE **87**(7) pp. 1062–1078 (1999)

 Petitcolas F.A.P.: MP3Stego (2006).
`http://www.petitcolas.net/fabien/steganography.Cited20Oct2008`

 Shirali-Shahreza M., Shirali-Shahreza M.H.: Text Steganography In SMS.
Int. Conference on Convergence Information Technology pp. 2260–2265 (2007)

 Simmons G.J.: The History of Subliminal Channels.
IEEE Journal on Selected Areas in Communications, **16**(4), pp. 452–462 (1998)

 Zhu W., Thomborson C.: Recognition in Software Watermarking.
Proceedings of the 4th ACM international workshop on Contents protection and security, pp. 29–36. ACM (2006)

Outline

Questions?

Thank you

Steganalysis of Hydan

IFIP Sec 2009 18-20 May, Paphos Cyprus

Jorge Blasco[1], Julio C. Hernández-Castro[1], Juan M.E. Tapiador[1], Arturo Ribagorda[1] and Miguel A. Orellana-Quirós[2]

[1]Carlos III University of Madrid, Computer Science Department
[2]Ministry of Economy, Madrid

May 18th, 2009