

security is not an island
HILTONMALTA

24th Annual **FIRST**
Conference
MALTA
17 - 22 June 2012



Cryptanalysis of malware encrypted output files

Nelson Uto

CPqD

A large, bold, black logo consisting of the letters 'CPqD' in a stylized, italicized font. The 'C' and 'P' are connected, and the 'q' and 'D' are also connected. The logo is positioned on the right side of the slide.

24th Annual **FIRST**
Conference

MALTA

17 - 22 June 2012

Agenda

- Introduction.
- Cryptanalysis of File #1.
- Cryptanalysis of File #2.
- Cryptanalysis of File #3.

Introduction

- CPqD was hired by a big Brazilian company to find out which information had been stolen by three different malwares, that infected its environment.
- Each one of them stored information in encrypted form using different mechanisms.
- We did only have access to the encrypted files and the malware binaries, meaning we could not use the special purpose hardware targeted by them.
- Due to the sensitivity of the stolen data and signed NDA, this talk will not use the real information we retrieved from those files.

Covered topics

- Detection of weak cryptosystems.
- Cryptanalysis of classical algorithms.
- Block ciphers.
- DES.
- Modes of operation.
- Searching key in malware binary or in memory.
- Worst scenario.

File #1 – Sample

#01.enc - GHex

Arquivo Editar Ver Janelas Ajuda

00000000	A6	B7	A7	69	94	89	AD	BB	B3	72	9E	A8	69	97	8E	AE	...	i.....r..i...
00000010	BE	C1	BF	B0	B6	B2	BD	8E	68	A2	94	95	A4	94	92	A2	h.....
00000020	79	68	6F	6F	72	6F	62	69	6E	40	68	6F	6F	72	6F	62	yhoorobin@hoorob	
00000030	69	6E	40	68	6F	6F	72	6F	62	69	6E	40	68	6F	6F	72	in@hoorobin@hoor	
00000040	6F	62	69	6E	89	BB	6F	B0	72	9B	A3	BB	B5	85	68	9C	obin..o.r...h.	
00000050	BE	C5	B0	AB	AC	6E	6F	AE	6F	A3	B7	B2	AA	B7	B7	83no.o.....	
00000060	A9	BB	7B	72	55	D7	6E	22	23	B5	28	9B	F6	E6	3A	59	..{rU.n"#.(...:Y	
00000070	D7	5A	E0	12	44	CD	4D	0A	31	DF	F2	4C	96	DE	3B	17	.Z..D.M.1..L...;	
00000080	38	7B	31	37	D4	06	23	F4	89	D4	E2	26	40	C2	1A	3A	8{17..#....&@...:	
00000090	3C	0A	41	FC	8A	6C	68	6F	6F	72	6F	62	69	6E	40	68	<.A..lhoorobin@h	
000000A0	6F	6F	72	6F	62	69	6E	40	68	6F	6F	72	6F	62	69	9E	oorobin@hoorobi.	
000000B0	92	B7	B2	B4	B6	C4	B4	AA	BA	40	89	BD	B3	72	9F	AA@...r..	
000000C0	C2	C1	89	AB	B0	BB	72	92	B1	B7	C2	92	B7	BB	C2	72r.....r	
000000D0	A3	AA	AA	C2	40	9B	B4	B4	BD	C2	62	9D	BD	40	98	C1@.....b..@..	
000000E0	BE	C6	B4	A5	BD	6E	74	B0	B4	6F	95	BE	B0	AF	B7	84nt..o.....	
000000F0	AD	BD	C3	BB	B0	AE	B2	C2	99	74	43	85	30	75	EE	79tC.0u.y	

8 bits com sinal: -90 32 bits com sinal: 1772599206 Hexadecimal: A6

8 bits sem sinal: 166 32 bits sem sinal: 1772599206 Octal: 246

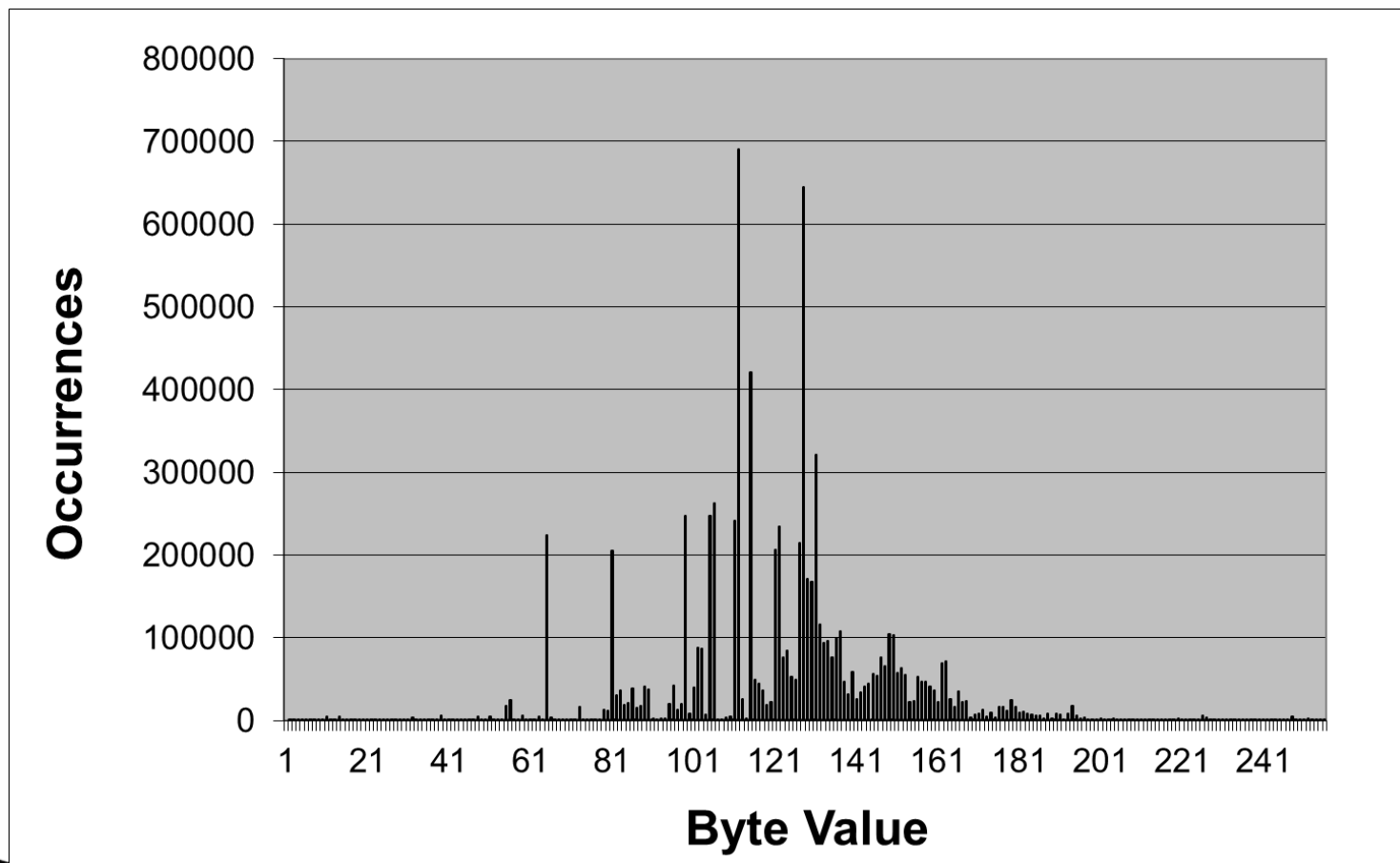
16 bits com sinal: -18522 Flutuante de 32 bits: 2.534473e+25 Binário: 10100110

16 bits sem sinal: 47014 Flutuante de 64 bits: -3.127396e-21 Tamanho do fluxo: 8

Mostrar decodificação little endian Mostrar números sem sinal e flutuantes como hexadecimal

Deslocamento: 0

File #1 – Histogram



File #1 – Important facts

- File#1 is pretty redundant.
 - This means a weak cryptosystem was used.
- The distance between occurrences of the string “robin@hoo” is always multiple of its length.
- Most of the bytes has values between 80 and 180.

File #1 – Hypothesis

- **Hypothesis #1:** a constant number is added to each byte modulo 256 and a given string is repeated several times in the plain text.
 - Not likely, but it should be tested.
 - How?
- **Hypothesis #2:** a Vigenère cipher over an alphabet of 256 elements and period equals 9 was used.
 - Candidate key: robin@hoo

File #1 – First attempt

#01.dec1 - GHex

Arquivo Editar Ver Janelas Ajuda

00000000	34	48	45	00	26	49	45	4C	44	00	2F	46	00	29	4E	46	4HE.&IELD./F.)NF
00000010	4F	52	4D	41	54	49	4F	4E	00	33	25	23	35	32	29	34	ORMATION.3%#52)4
00000020	39	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	9.....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000040	00	00	00	00	49	53	00	41	00	2C	41	52	47	45	00	2DIS.A.,ARGE.-
00000050	4F	53	41	49	43	00	2F	46	00	34	45	43	48	4E	49	43	OSAIC./F.4ECHNIC
00000060	41	4C	0C	00	E6	75	05	B4	E3	4D	B9	2C	84	77	D8	F0	AL...u...M.,.w..
00000070	69	1A	78	A3	D5	5B	DE	A8	C8	71	B2	E4	27	6F	C9	A8	i.x..[...q..'o..
00000080	D6	12	C3	F7	6C	97	B4	82	1A	72	79	B8	00	5A	AB	CBl....ry..Z..
00000090	CA	9B	DF	93	1C	2C	00	00	00	00	00	00	00	00	00	00,
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	300
000000B0	52	4F	43	45	44	55	52	41	4C	00	21	4E	44	00	30	48	ROCEDURAL.!ND.0H
000000C0	59	53	49	43	41	4C	00	23	4F	4E	54	52	4F	4C	53	00	YSICAL.#ONTROLS.
000000D0	34	48	41	54	00	33	45	45	4B	53	00	34	4F	00	30	52	4HAT.3EEKS.40.0R
000000E0	4F	54	45	43	54	00	34	48	45	00	23	4F	4E	46	49	44	OTECT.4HE.#ONFID
000000F0	45	4E	54	49	41	4C	49	54	59	0C	D4	16	BE	06	8C	10	ENTIALITY.....

8 bits com sinal: 52 32 bits com sinal: 4540468 Hexadecimal: 34

8 bits sem sinal: 52 32 bits sem sinal: 4540468 Octal: 064

16 bits com sinal: 18484 Flutuante de 32 bits: 6.362551e-39 Binário: 00110100

16 bits sem sinal: 18484 Flutuante de 64 bits: 2.672255e+59 Tamanho do fluxo: 8

Mostrar decodificação little endian Mostrar números sem sinal e flutuantes como hexadecimal

Deslocamento: 0

File #1 – Correction

#01.dec2 - GHex

Arquivo Editar Ver Janelas Ajuda

00000000	54	68	65	20	46	69	65	6C	64	20	4F	66	20	49	6E	66	The Field Of Inf
00000010	6F	72	6D	61	74	69	6F	6E	20	53	45	43	55	52	49	54	ormation SECURIT
00000020	59	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	Y
00000030	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	
00000040	20	20	20	20	69	73	20	61	20	4C	61	72	67	65	20	4D	is a Large M
00000050	6F	73	61	69	63	20	4F	66	20	54	65	63	68	6E	69	63	osaic Of Technic
00000060	61	6C	2C	20	06	95	25	D4	03	6D	D9	4C	A4	97	F8	10	al, ..%.m.L....
00000070	89	3A	98	C3	F5	7B	FE	C8	E8	91	D2	04	47	8F	E9	C8{.....G...
00000080	F6	32	E3	17	8C	B7	D4	A2	3A	92	99	D8	20	7A	CB	EB	.2.....:.... z...
00000090	EA	BB	FF	B3	3C	4C	20	20	20	20	20	20	20	20	20	20<L
000000A0	20	20	20	20	20	20	20	20	20	20	20	20	20	20	20	50	P
000000B0	72	6F	63	65	64	75	72	61	6C	20	41	6E	64	20	50	68	rocedural And Ph
000000C0	79	73	69	63	61	6C	20	43	6F	6E	74	72	6F	6C	73	20	ysical Controls
000000D0	54	68	61	74	20	53	65	65	6B	73	20	54	6F	20	50	72	That Seeks To Pr
000000E0	6F	74	65	63	74	20	54	68	65	20	43	6F	6E	66	69	64	otect The Confid
000000F0	65	6E	74	69	61	6C	69	74	79	2C	F4	36	DE	26	AC	30	entiality,.6.&.0

8 bits com sinal: 84 32 bits com sinal: 543516756 Hexadecimal: 54

8 bits sem sinal: 84 32 bits sem sinal: 543516756 Octal: 124

16 bits com sinal: 26708 Flutuante de 32 bits: 1.943157e-19 Binário: 01010100

16 bits sem sinal: 26708 Flutuante de 64 bits: 1.441612e+214 Tamanho do fluxo: 8

Mostrar decodificação little endian Mostrar números sem sinal e flutuantes como hexadecimal

Deslocamento: 0

File #1 – Description of cipher

- **Alphabet of definition:** $\mathcal{A} = \{0, 1, 2, 3, \dots, 255\}$
- **Plain text:** $M = m_0m_1m_2\dots m_{t-1}, m_i \in \mathcal{A}$
- **Cipher text:** $C = c_0c_1c_2\dots c_{t-1}, c_i \in \mathcal{A}$
- **Key:** $K = k_0k_1k_2k_3k_4k_5k_6k_7k_8$
 $= 0x52\ 4f\ 42\ 49\ 4e\ 20\ 48\ 4f\ 4f$
- **Encryption function:** $c_i = m_i + k_{(i \bmod 9)} \bmod 256$
- **Decryption function:** $m_i = c_i - k_{(i \bmod 9)} \bmod 256$

File #2 – Sample

```
esruser@ubuntu: /tmp
Arquivo Editar Ver Terminal Ajuda
esruser@ubuntu:/tmp$ cat \#02.enc
iPlKR5LehJf6FP4sWSDmQvY07PcZjZi5WSvk287c2/UU5N2mC+vagZvA7LVuJZm4+UMyAlDUwZDqFXKC
3GcMBeyAnRw/ fi1WX7UpAM0VU8Pb0op8yMTYw6w9E06xc f84Zrduknf2B54=8KmMOBLFRQM7jGzCWhGv
1wt79lX0c0FNc7DDGqKu31Y=6LDPjUYL77UjPcCYB5KoVEcNnoMRB7dHFYAfPP7xl64aRRquDDjwcPcu
Awq97cpwpThzD0GZQww9n66rnFkuS8kZ35Gjzm68RYGeRHdLQrU=napjM3ySbBAHhs3XQub+uh/Gbn6W
rCKs+oqXXWgdLg0=q17TBIooNpbFCDxKVp3D7WvF2Zp8Vzg5mcbcjEhzH7cwLz9eEo/o0gCZfH4xJTmn
2b//uSpLcKwz3bVBZ9FBdNHERICThgTbzu/buXDSM5Q=Cm0mIiwcR6MxFsRoEtw0SUTZpVLardwtd9U8
Qoc3TK2tKQd4ybR2jsawGhWb5FKQ1eLYnQnxQm0wuf7r0jTLKWNcU8w65V/QJnttWl6umYLGCGVa/3
I4CG6N2yBNssv9GN1ig0B60=NcSrmv7CWtuSg1Lr7xhodbpffhsSLwqyJTqUhKjSGwcWPVN5aqa2CT1g
w+Adv2ERx6YBo0s1c60cFVVYTetB9BBWda6QPvriTVi2jy9av8=s88S2fScw6j14DeS+e6f/OSjhEAU
W79h8KNrNKomocybmRPXmL0v9A==KtY0/kFWbjhYvyw7S/+4qEkHD7CtQT16MTK4feCHE2bZv/+5Kktw
rJ4/KNtu0uiUi1/CXv6pmDVCd0F4hEePCyGHqZg0Nr74VJ8STg8r6xE5Rfkyxb50ALSN7BFevkMGckn
PmBMZt8=uQaZXZJBi3Bzn8Wq9idlGFw/YFjcjixaHpbqfZEPbFqq25Tg7lH0eQHDbh0+EZ/MP7PPS7bY
k7KeuE2pNmG3jQ==7xUECpPc+BRLCoAIowm1v53CxdNuTTvHxwY5swFN/5YBs0z0ci14ySDtMMQfQIZ
Rmg4k9W5oZeBjVm01JoD08X4eq4CU71cl32K1R6q24s3Mu7B5mpDuZ8rHGxgMJCv06zI+BHiudg=ce3p
+chcBF4j6r3S62ZhJotxw6dyPNheNW/MZA8J2uFZ28+Z/BAC9CmQrSVap/vzkYH/Np42Igo=EMIWBaVd
hGKQXhn0P1cj2kl2dCrUrRlKQ0bhxlmaxLB08nWGF6LKDZ1Rj3oGt4SiuVFBo7+qMKy5rVe01PcLtfEa
qqjqKMygKHKW+WQ64iSfHSpjGmx5WqV4UgIdAk6zzCoVDxE74Kg=Gu0ykJlEh6Eo/04YvT3RaRuP9EG6
tDKC0Ut7BZD6qn/zNjpcgafW86btFD1yQ4U0s5LYeqvo6g4n02xgQchLG18Vk0lKca/l3yauFS75S0HZ
ypK3JMFhwIHft6ezQgqaSGN6BHydViL7+byddhxkSjVI9LrSrdokKeAJQEAnblvJ4fAtpolL7csXnDUT
XXjQruicjP0tH3CAqowP43pm00/7BxDFahN2l7aJ4HV2ly0cum46dLLtfw==jRk2ZM/mHKNEwNSNMQnC
F1IHTCT9CrSqMKNia5p3h1Cwlrdp8r1AqA==HoQDgALw5wH6aBd4pFGHMGSJ2HrYGCmWo0DuNME8PjU=
nhm70YsfD0FQF3tPj rR+SAHbMNPLK4r7+0235tGnJ6M=pKDYE0nJANyKBSKH27D1haKnNJGzT30yH038
KcCBunsHbpqGruwLJQmGuPNsq32/WSvk287c2/U=fKkTSiBLVVDpoNU/9g+U8FSlK1cT++idbRUQ344a
```

File #2 – Base64 decoded

#02.decoded - GHex

Arquivo Editar Ver Janelas Ajuda

```
00000000 88 F9 4A 47 92 DE 84 97 FA 14 FE 2C 59 20 E6 42 ..JG.....,Y .B
00000010 F6 34 EC F7 19 8D 98 B9 59 2B E4 DB CE DC DB F5 .4.....Y+.....
00000020 14 E4 DD A6 0B EB DA 81 9B C0 EC B5 6E 25 99 B8 .....n%..
00000030 F9 43 32 02 50 D4 C1 90 EA 15 72 82 DC 67 0C 05 .C2.P.....r..g..
00000040 EC 80 9D 1C 3F 7E 2D 56 5F B5 29 00 CD 15 53 C3 ....?~-V_.)...S.
00000050 DB D2 8A 7C C8 C4 D8 C3 AC 3D 10 EE B1 71 FF 38 ...|.....=...q.8
00000060 66 B7 6E 92 77 F6 07 9E F0 A9 8C 38 12 C5 45 03 f.n.w.....8..E.
00000070 3B 8C 6C C2 5A 11 AF D7 0B 7B F6 55 F4 73 41 4D ;.l.Z....{.U.sAM
00000080 73 B0 C3 1A A2 AE DF 56 E8 B0 CF 8D 46 0B EF B5 s.....V....F...
00000090 23 3D C0 98 07 92 A8 54 47 0D 9E 83 11 07 B7 47 #=.....TG.....G
000000A0 15 80 1F 3C FE F1 97 AE 1A 45 1A AE 0C 38 F0 70 ...<.....E...8.p
000000B0 F7 2E 03 0A BD ED CA 70 A5 38 73 0C E1 99 43 0C .....p.8s...C.
000000C0 3D 9F AE AB 9C 59 2E 4B C9 19 DF 91 A3 CC CE BC =....Y.K.....
000000D0 45 81 9E 44 77 4B 42 B5 9D AA 63 33 7C 92 6C 10 E..DwKB...c3|.l.
000000E0 07 1E CD D7 42 E6 FE BA 1F C6 6E 7E 96 AC 22 AC ....B.....n~..".
000000F0 FA 8A 97 5D 68 1D 2E 0D AB 5E D3 04 8A 28 36 96 ...]h....^...(6.
```

8 bits com sinal: -120 32 bits com sinal: 1196095880 Hexadecimal: 88
8 bits sem sinal: 136 32 bits sem sinal: 1196095880 Octal: 210
16 bits com sinal: -1656 Flutuante de 32 bits: 5.196153e+04 Binário: 10001000
16 bits sem sinal: 63880 Flutuante de 64 bits: -2.233486e-195 Tamanho do fluxo: 8

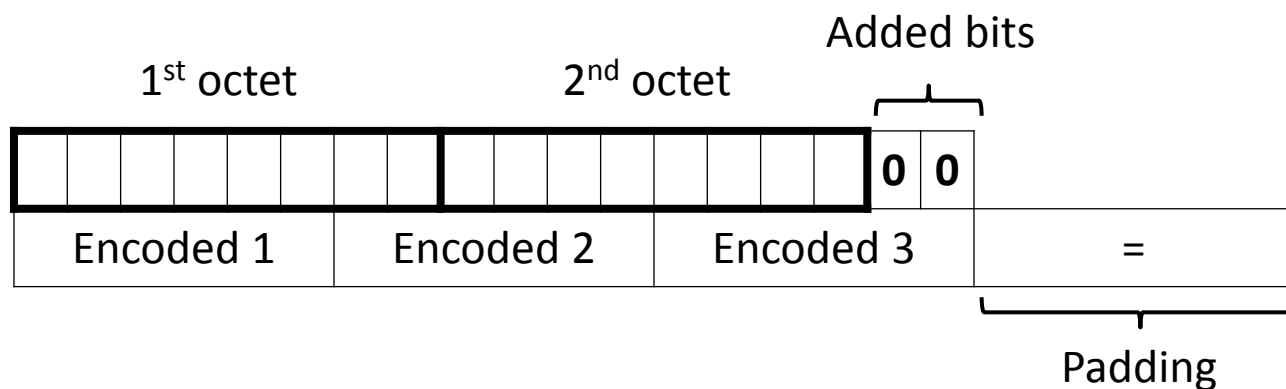
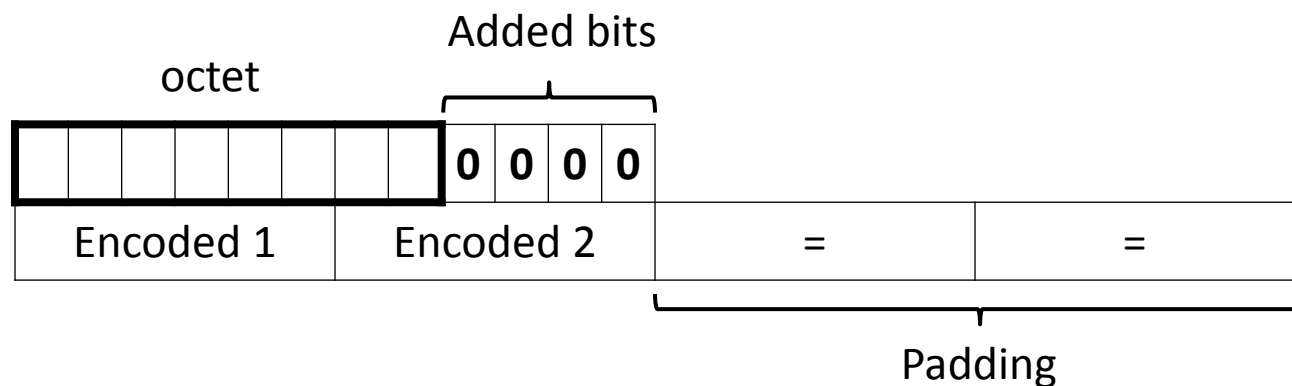
Mostrar decodificação little endian Mostrar números sem sinal e flutuantes como hexadecimal

Deslocamento: 0

File #2 – Redundancy check

```
esruser@ubuntu: /tmp
Arquivo Editar Ver Terminal Ajuda
esruser@ubuntu:/tmp$ ls -l \#02.decoded
-rw-r--r-- 1 esruser esruser 2032 2012-06-07 11:12 #02.decoded
esruser@ubuntu:/tmp$ gzip \#02.decoded
esruser@ubuntu:/tmp$ ls -l *gz
-rw-r--r-- 1 esruser esruser 2067 2012-06-07 11:12 #02.decoded.gz
esruser@ubuntu:/tmp$
```

File #2 – Base64 review



File #2 – Block size?

```
esruser@ubuntu: /tmp
Arquivo Editar Ver Terminal Ajuda
esruser@ubuntu:/tmp$ cat \#02.enc
iPlKR5LehJf6FP4sWSDmQvY07PcZjZi5WSvk287c2/UU5N2mC+vagZvA7LVuJZm4+UMyA1DUwZDqFXKC
3GcMBeyAnRw/ fi1WX7UpAM0VU8Pb0op8yMTYw6w9E06xc f84Zrduknf2B54=8KmMOBLFRQM7jGzCWhGv
1wt79lX0c0FNc7DDGqKu31Y=6LDPjUYL77UjPcCYB5KoVEcNnoMRB7dHFYAfPP7x164aRRquDDjwcPcu
Awq97cpwpThzD0GZQww9n66rnFkuS8kZ35Gjz M68RYGeRHdLQRu=napjM3ySbBAHhS3XQub+uh/Gbn6W
rCKs+oqXXWgdLg0=q17TBIooNpbFCDxKVp3D7WvF2Zp8Vzg5mcbcjEhzH7cwLz9eEo/o0gCZfH4xJTmn
2b//uSpLcKwz3bVBZ9FBdNHERICThgTbzbuXDSM5Q=Cm0mIiwcR6MxFsRoEtw0SUTZpVLardwtd9U8
Qoc3TK2tKQd4ybR2jsawGhWb5FKQ1eLYnQnxQm0wuf7r0jTLKWnCU8w65V/QJnttWl6umYLGCGVa/3
I4CG6N2yBNssv9GN1ig0B60=NcSrmv7CWtuSg1Lr7xhodbpffhsSLwqyJTqUhKjSGwcWPVN5aqa2CT1g
w+Adv2ERx6YBo0s1c60cFVVYTetB9BBWda6QPVriTVi2jy9av8=s88S2fScw6j14DeS+e6f/OSjhEAU
W79h8KNrNKomocybmRPXmL0v9A==KtY0/kFWbjhYvyw7S/+4qEkHD7CtQT16MTK
rJ4/KNtu0uiUi1/CXv6pmDVCd0F4hEePCyGHqZg0Nr74VJ8STg8r6xE5Rfkyxb
PmBMZt8=uQaZXZJBi3Bzn8Wq9idlGFw/YFjcjixaHpbqfZEPbFqq25Tg7lH0eQH
k7KeuE2pNmG3jQ==7xUECpPc+BRLCoAIowm1v53CxdNuTTvHxwY5swFN/5YBsG
Rmg4k9W5oZeBjVm01JoD08X4eq4CU71cl32K1R6q24s3Mu7B5mpDuZ8rHGxgMJC
+chcBF4j6r3S62ZhJotxw6dyPNheNW/MZA8J2uFZ28+Z/BAC9CmQrSVap/vzkYH
hGKQXhn0P1cj2kl2dCrUrRlKQ0bhxLmaxLB08nWGF6LKDZ1Rj3oGt4SiuVFB07+
qqjKMygKHKW+WQ64iSfHSpjGmx5WqV4UgIdAk6zzCoVDxE74Kg=Gu0ykJ1eh6E
tDKC0Ut7BZD6qn/zNjpcgafW86btFD1yQ4U0s5LYeqvo6g4n02xgQchLG18Vk0Lkca/
c5yaur3755qnz
ypK3JMFhwIHft6ezQggaSGN6BHydViL7+byddhxkSjVI9LrSrdokKeAJQEAnlvJ4fAtpoLL7csXnDUT
XXi0ruiciP0tH3CAaowP43pm00/7BxDFahN217aJ4HV2ly0cum46dLLtfw==jRk2ZM/mHKNEwNSNMQnC
F1IHTCT9CrSqMKNia5p3h1CWlrdp8rLAqa==loQDgALw5wH6aBd4pFGHMGSJzHrYgcmwoouDUNME8PJU=
nhm/OYstD0FQF3tPjRr+SAHbMNPLK4r/+0235tGnJ6M=pKDYE0nJANyKBSKH27D1haKnJGzT30yH038
KcCBunsHbpqGruwLJQmGuPNsq32/WSvk287c2/U=fKkTSiBLVVDpoNU/9g+U8FSlK1cT++idbRUQ344a
```

- 1) Length = 56 Base64 chars.
- 2) Ends with “==”.
- 3) Therefore input length equals 40 bytes.
- 4) Possible block size: 64 bits.

File #2 – Candidate ciphers

- DES.
- 2TDES.
- 3TDES.
- FEAL.
- IDEA.
- SAFER.
- RC5.
- LOKI.
- Blowfish.

File #2 – String search

```
esruser@ubuntu: /tmp
Arquivo Editar Ver Terminal Ajuda
esruser@ubuntu:/tmp$ strings Portsys.exe.malware | grep -i "encrypt"
esruser@ubuntu:/tmp$ strings Portsys.exe.malware | grep -i "crypto"
esruser@ubuntu:/tmp$ strings Portsys.exe.malware | grep -i "cipher"
ECipherException
LbCipher
esruser@ubuntu:/tmp$ strings Portsys.exe.malware | grep -i "DES"
IDesignerNotify
DesignSize
IDesignerHook,(A
poDesigned      poDefault
poDesktopCenter
      dmDesktop      dmPrimary
      OnDestroyT
GetDesktopWindow
DestroyWindow
DestroyMenu
DestroyIcon
DestroyCursor
ImageList_Destroy
esruser@ubuntu:/tmp$ strings Portsys.exe.malware | grep -i "bf"
esruser@ubuntu:/tmp$ strings Portsys.exe.malware | grep -i "blowfish"
esruser@ubuntu:/tmp$
```

File #2 – Narrowing the options

- LbCipher is a library for Delphi.
- It implements the following algorithms from our list:
 - Blowfish (ECB, CBC).
 - DES (ECB, CBC).
 - 2TDES (ECB, CBC).
 - 3TDES (ECB, CBC).

File #2 – Starting with DES

- DES is a 64-bit block cipher.
- The cipher employs a 64-bit key of which only 56 bits are effective.
- Based on a Feistel network.
- It is possible to search the entire key space using special purpose hardware¹, which was first built in 1998.

File #2 – Inside DES (1)

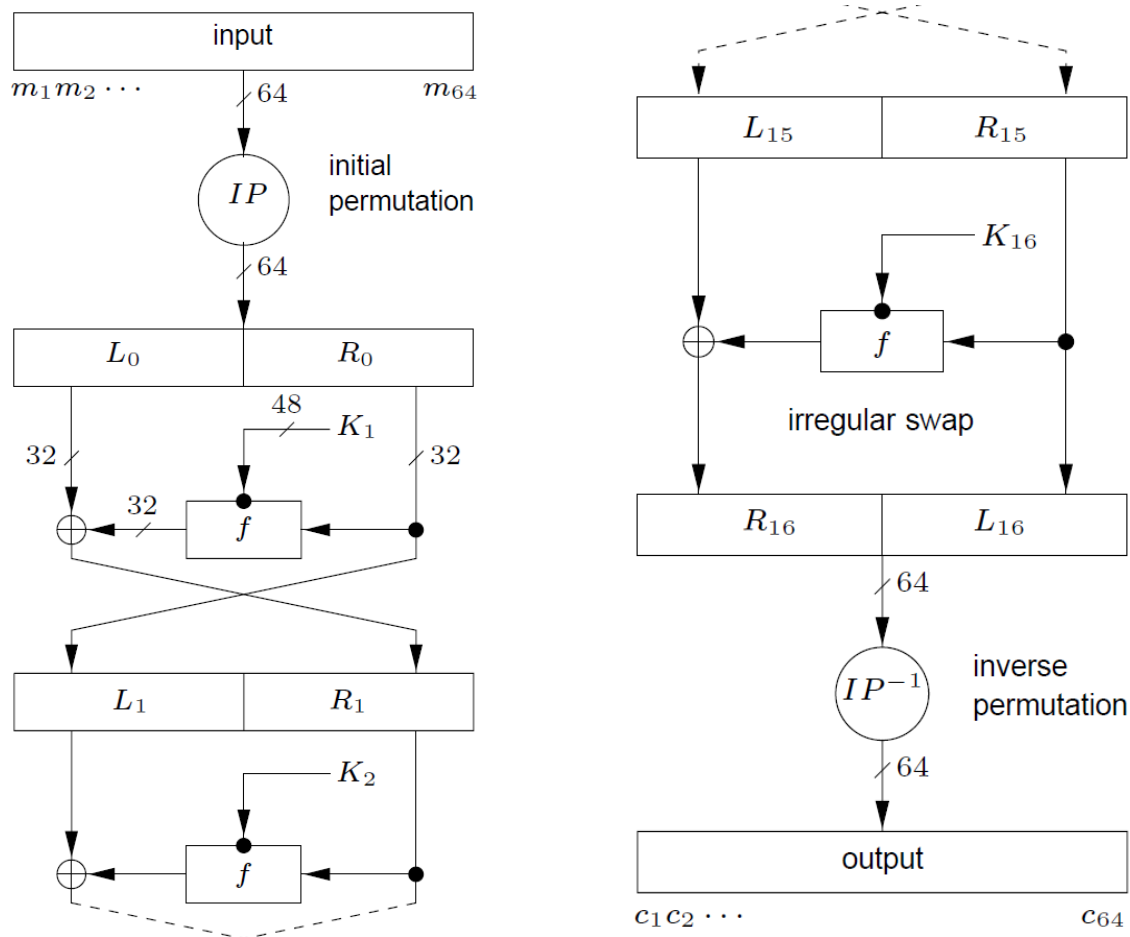


Figure: DES rounds.

Source: [2] HAC.

File #2 – Inside DES (2)

IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Figure: DES initial permutation and inverse.

Source: [2] HAC.

File #2 – Inside DES (3)

<i>E</i>					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

<i>P</i>			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Figure: DES round function expansion E and permutation P.

Source: [2] HAC.

File #2 – Inside DES (4)

PC1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
above for C_i ; below for D_i						
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

PC2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Figure: DES key schedule bit selections.

Source: [2] HAC.

File #2 – From LbCipher

```
procedure InitEncryptDES(const Key : TKey64;  
    var Context : TDESContext; Encrypt : Boolean);  
const PC1 : array [0..55] of Byte = (56, 48, 40, 32, 24, 16, 8, 0,  
57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18, 10, 2, 59, 51,  
43, 35, 62, 54, 46, 38, 30, 22, 14, 6, 61, 53, 45, 37, 29, 21, 13, 5,  
60, 52, 44, 36, 28, 20, 12, 4, 27, 19, 11, 3);  
PC2 : array [0..47] of Byte = (13, 16, 10, 23, 0, 4, 2, 27, 14, 5, 20,  
9, 22, 18, 11, 3, 25, 7, 15, 6, 26, 19, 12, 1, 40, 51, 30, 36, 46, 54,  
29, 39, 50, 44, 32, 47, 43, 48, 38, 55, 33, 52, 45, 41, 49, 35, 28,  
31);
```

File #2 – Next steps

- Load the malware in OllyDbg.
- Search for PC1 and use it to locate the address of InitEncryptDES, if present.
- Set a breakpoint in that address.
- Run the malware.
- Extract the key from the first parameter.

File #2 – Finding PC1 (1)

Address	Hex dump	ASCII	
00451000	00 00 00 00 00 00 00 00	
00451008	02 8D 40 00 00 00 00 00	□□@.....	
00451010	00 00 00 00 00 00 00 00		
00451018	00 00 00 00 00 00 00 00		
00451020	32 13 8B C0 02 00 00 00		
00451028	00 8D 40 00 00 00 00 00		
00451030	00 8D 40 00 01 80 00 00		
00451038	00 00 00 00 00 00 00 00		
00451040	28 21 40 00 B8 20 00 00		
00451048	38 26 40 00 00 C0 00 00		
00451050	C9 D7 CF C8 CD CE DB D8	E×IEIIUØ	
00451058	DA D9 CA DC DD DE DF E0	ÚÛÊÛÝÞßà	
00451060	E1 E3 00 E4 E5 8D 40 00	áã.ää□@.	
00451068	45 72 72 6F 72 00 8B C0	Error.<À	
00451070	52 75 6F 74 69 6D 65 20	Runtime	

Enter binary string to search for

ASCII: 80< ↵

UNICODE:

HEX +06: 38 30 28 20 18 10

Entire block
 Case sensitive

<< >> OK Cancel

File #2 – Finding PC1 (2)

Address	Hex dump	ASCII
00451E48	38 30 28 20 18 10 08 00	80(.
00451E50	39 31 29 21 19 11 09 01	91)!.
00451E58	3A 32 2A 22 1A 12 0A 02	:2*".
00451E60	3B 33 2B 23 3E 36 2E 26	;3+#>6.&
00451E68	1E 16 0E 06 3D 35 2D 25	====5-8
00451E70	1D 15 0D 05 3C 34 2C 24	..<4,\$
00451E78	1C 14 0C 04 1B 13 0B 03	..
00451E80	0D 10 0A 17 00 04 02 1B	. . .
00451E88	0E 05 14 09 16 12 0B 03	. . .
00451E90	19 07 0F 06 1A 13 0C 01
00451E98	28 33 1E 24 2E 36 1D 27	(3\$.6'
00451EA0	32 2C 20 2F 2B 30 26 37	2, /+0&7
00451EA8	21 34 2D 29 31 23 1C 1F	!4-)1#
00451EB0	01 02 04 06 08 0A 0C 0E
00451EB8	0F 11 13 15 17 19 1B 1C

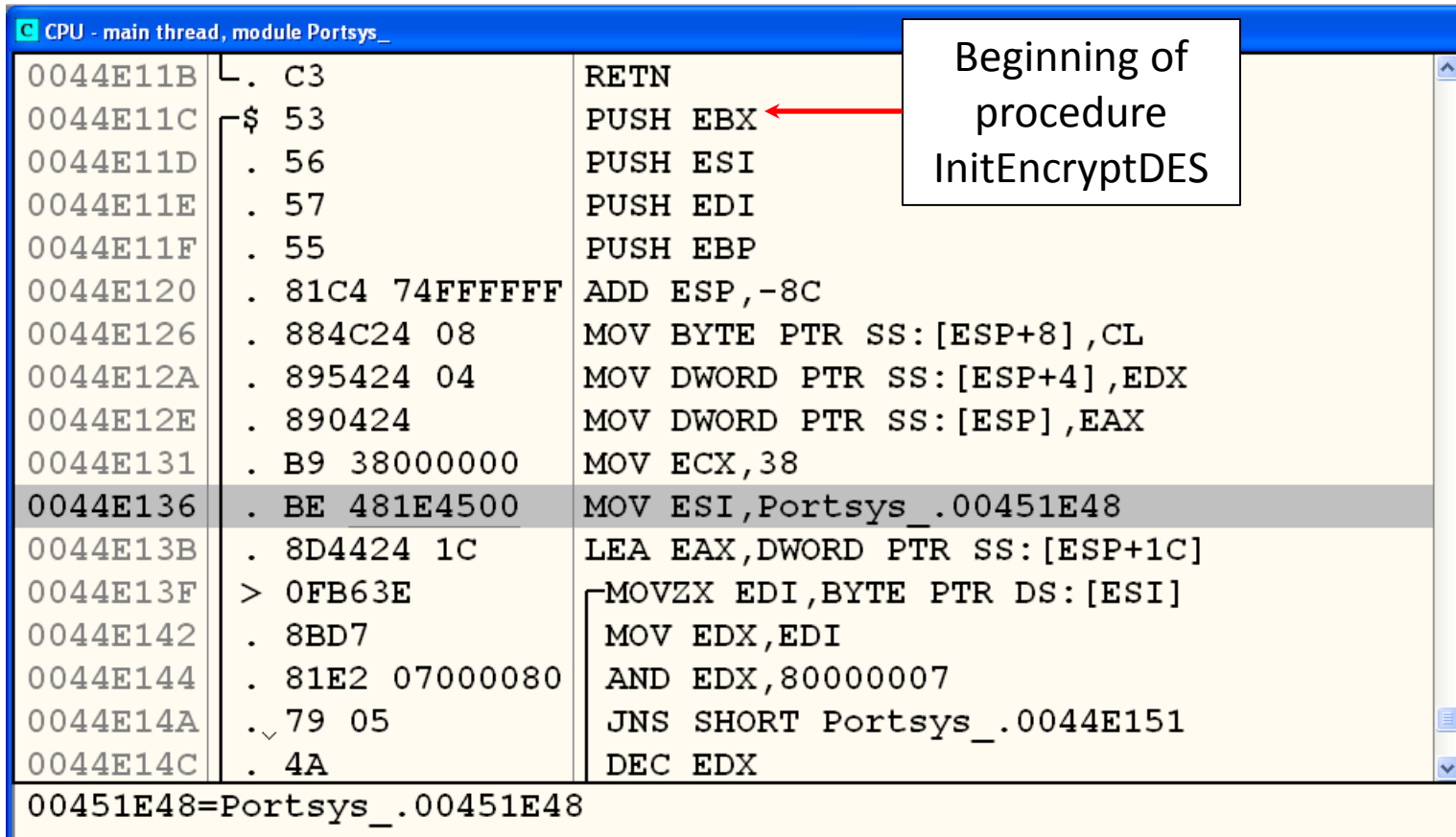
File #2 – References

Address	Hex dump	ASCII		0012FFC4	7C817077	F
00451E48	38 30 28 20 18 10 08 00	80 (□□□.		0012FFC8	00000001	
00451E50	39 31 29 21 19 11 09 01	91)!□□.□		0012FFCC	00000000	

Address	Disassembly	Comment
0044E136	MOV ESI,Portsys_.00451E48	00451E48=Portsys_.00451E48

To find references to PC1, we need to select its first byte (0x38) and press Ctrl+R.

File #2 - Beginning of the function



```
CPU - main thread, module Portsys_
0044E11B  .  C3          RETN
0044E11C  $  53          PUSH EBX
0044E11D  .  56          PUSH ESI
0044E11E  .  57          PUSH EDI
0044E11F  .  55          PUSH EBP
0044E120  .  81C4 74FFFFFF ADD ESP,-8C
0044E126  .  884C24 08     MOV BYTE PTR SS:[ESP+8],CL
0044E12A  .  895424 04     MOV DWORD PTR SS:[ESP+4],EDX
0044E12E  .  890424       MOV DWORD PTR SS:[ESP],EAX
0044E131  .  B9 38000000  MOV ECX,38
0044E136  .  BE 481E4500  MOV ESI,Portsys_.00451E48
0044E13B  .  8D4424 1C     LEA EAX,DWORD PTR SS:[ESP+1C]
0044E13F  > 0FB63E       MOVZX EDI,BYTE PTR DS:[ESI]
0044E142  .  8BD7         MOV EDX,EDI
0044E144  .  81E2 07000080 AND EDX,80000007
0044E14A  .  79 05        JNS SHORT Portsys_.0044E151
0044E14C  .  4A          DEC EDX
00451E48=Portsys_.00451E48
```

File #2 – Running the malware

The screenshot shows a debugger window titled "CPU - main thread, module Portsys_". The main window displays assembly code with addresses, hex values, and instructions. A context menu is open over the instruction at address 0044E12E, with the "Breakpoint" option selected. A sub-menu is also open, showing options like "Toggle", "Hit trace", "Run to selection", etc.

Address	Hex	Instruction
0044E11B	. C3	RETN
0044E11C	\$. 53	PUSH EBX
0044E11D	. 56	PUSH ESI
0044E11E	. 57	PUSH EDI
0044E11F	. 55	PUSH EBP
0044E120	. 81C4 74FFFFFF	ADD ESP, -8C
0044E126	. 884C24 08	MOV BYTE PTR SS:[ESP+8], CL
0044E12A	. 895424 04	MOV DWORD PTR SS:[ESP+4], EDX
0044E12E	. 890424	MOV DWORD PTR SS:[ESP], EAX
0044E131	. .	MOV ECX, 38
0044E136	. .	MOV ESI, Portsys_.00451E48
0044E13B	. .	LEA EAX, DWORD PTR SS:[ESP+1C]
0044E13F	> .	MOV EAX, DWORD PTR DS:[ESI]
0044E142	. .	
0044E144	. .	
0044E14A	. .	
0044E14C	. .	

Address	Hex	ASCII
00451E48	38	08 00 80 (□□□.
00451E50	39	09 01 91) !□□.□
00451E58	3A	0A 02 :2*"□□.□

File #2 – Which parameter?

- Remember the procedure signature is as follows:

```
procedure InitEncryptDES(  
    const Key : TKey64;  
    var Context : TDESContext;  
    Encrypt : Boolean);
```
- TKey64 definition:

```
TKey64 = array [0..7] of Byte;
```
- A TKey64 value can not be stored by a single register in a 32-bit architecture.

File #2 – Calling convention

- Delphi's calling convention (left-to-right):
 - 1st parameter: EAX.
 - 2nd parameter: EDX.
 - 3rd parameter: ECX.
 - Remaining parameters: stack.

File #2 – Key address

```
Registers (FPU) < <
EAX 00453C04 Portsys_.00453C04
ECX 00453C01 Portsys_.00453C01
EDX 0012FB4B
EBX 009877BC
ESP 0012FB38
EBP 0012FBD8
ESI 009877EC
EDI 00412430 Portsys_.00412430
EIP 0044E11C Portsys_.0044E11C
C 1 ES 0023 32bit 0 (FFFFFFFF)
P 1 CS 001B 32bit 0 (FFFFFFFF)
A 0 SS 0023 32bit 0 (FFFFFFFF)
Z 0 DS 0023 32bit 0 (FFFFFFFF)
S 0 FS 003B 32bit 7FFDF000 (FFF)
T 0 GS 0000 NULL
D 0
```

File #2 – Key value

Address	Hex dump	ASCII
00453C04	C2 4F A0 10 74 4E B1 53	ÂO □tN±S
00453C0C	FF FF FF FF 00 00 00 00	ÿÿÿÿ....
00453C14	00 00 00 00 00 00 00 00
00453C1C	00 00 00 00 00 00 00 00
00453C24	00 00 00 00 00 00 00 00
00453C2C	00 00 00 00 00 00 00 00
00453C34	00 00 00 00 00 00 00 00
00453C3C	00 00 00 00 00 00 00 00
00453C44	00 00 00 00 00 00 00 00
00453C4C	00 00 00 00 00 00 00 00
00453C54	00 00 00 00 00 00 00 00
00453C5C	00 00 00 00 00 00 00 00
00453C64	00 00 00 00 00 00 00 00
00453C6C	00 00 00 00 00 00 00 00
00453C74	00 00 00 00 00 00 00 00

File #2 – Description of cipher

- **Encryption algorithm:** DES.
- **Mode of operation:** ECB.
- **Key:** $K = 0xc24fa010744eb153$

Alternative for finding keys

- A properly generated key is entropic.
- Information, on the other hand, is structured.
- Based on those facts, in 1999, Shamir and Someren³ proposed a way of finding stored keys.
- The basic idea is to traverse memory and identify the region with more entropy.
- One way of doing that is to set a window size and count the number of different elements on each window.

File #3 – Sample

50E96823#0851CDA207333E24 1.0.6 St - P: 6 R: 11

CFT:1.0.2

PA: 3

C3@158BF7627CD2750FF53D7288C863F7C7041221CD8E77B6A7F7833815075091A23EB3ADA
2352ADFE9514952DE6DF8B619D41E51DFB7C0196A104F994920E2434716699DEF0DA48E624
CEC0953F7BE159E0B43F3862C4A8D8FE1476F7939F72F99A049CAC2DC1DE0E6BB91066FF3E9
20283A373E8B94DF3D39F06FCB6A29B9E5DCF20A0D02DE8F288F5C2737D1D64E1E25AA51A
42C0AAE3ABFE354EBCE781342A6D84413391F4038EDB213AA87870D25FC06DD05DBF3EEB6
84665A7E20C080F196BA42D96CFE0FA08FF64FF9B3C08CA3765768EDCBEDF620562ADB442C
6A1191A1A137E50C7F75C629AEB702F09F81107

PF: 3

50E96832#K@881A6DC9E4470F

50E96837#K@06BB

50E9683C#K@3FE759EE

File #3 – Description of cipher

- **Alphabet of definition:** $\mathcal{A} = \{0, 1, 2, 3, \dots, 255\}$
- **Plain text:** $M = m_0m_1m_2\dots m_{t-1}, m_i \in \mathcal{A}$
- **Cipher text:** $C = c_0c_1c_2\dots c_{t-1}, c_i \in \mathcal{A}$
- **Key:** $K = k_0k_1k_2k_3k_4k_5k_6k_7k_8k_9k_{10}$
- **Encryption function:** $c_i = m_i + k_{(i \bmod 11)} \bmod 256$
- **Decryption function:** $m_i = c_i - k_{(i \bmod 11)} \bmod 256$

References

- [1] Electronic Frontier Foundation, *Cracking Des: Secrets of Encryption Research, Wiretap Politics & Chip Design*, O'Reilly Media, 1998.
- [2] Menezes, A., van Oorschot, P, and Vanstone, S., *Handbook of Applied Cryptography*, CRC Press, 2001.
- [3] Shamir, A. and van Someren, N., *Playing "Hide and Seek" with Stored Keys*, in FC'99 Proc. of the 3rd Intl. Conference on Financial Cryptography, 1999.

Thank you for listening!
Questions?

Nelson Uto

uto@cpqd.com.br