Detection and Operational Cryptanalysis of Weakly Implemented, Weak or Trapped Encryption Systems A Step-by-Step Tutorial - Part I

Eric Filiol, filiol@esiea.fr

ESIEA - Laval
Operational Cryptology and Virology Lab $(C+V)^O$
http://www.esiea-recherche.eu/

H2HC 2010 - Sao Paulo & Cancun

Theoretical Crypto vs Real Crypto

- Secret key size is very often considered as a "key" security feature.
- Blind faith in cryptographic design.
  - "AES-256 inside" marketing syndrom.
  - Necessary but not sufficient condition.
- But what about implementation flaws ?
- Worse, what about intended trapdoors ?
- What about cryptographic misuses ?
  - Crypto has been deregulated but users never educated.
  - Confidence in cryptographic software can turn against users.
- Stream ciphers are still mainly used for sensitive traffics (e.g. perfect secrecy of Vernam ciphers).
  - What is the impact of key misuses or encryption algorithm (e.g. message key generator module) partial failure ?

What are the Issues ?

- Dual issues of security.
- On the user's side, the aim is
  - to detect implementation flaws or trapdoors,
  - without performing reverse-engineering (hard or soft) because it is horribly time-consuming and illegal !
- On the attacker's side, the aim is
  - to detect and break any weak traffic,
  - under the assumption that the cryptographic algorithm can be/remain unknown (e.g. satellite communications) !
- This talk presents an operational solution to all these issues.
- Method developped by the author in 1994.

Existing Works

- NSA Venona Project (1943 - 1980) to break the Soviet telex traffic.
  - Revealed by Peter Wright in 1987.
  - The method and ciphertexts still classified nowadays.
- E. Dawson & L. Nielsen (1996). Very empiric study. Detection is not addressed.
- J. Mason & al. (2006).
  - Detection is not addressed.
  - Very limited scope (file type must be known) and approach.
  - Complex method (HMM-based).
  - Really implemented ?
- Neither practical cases nor feedback on real cases addressed (just academic stuff).

Summary of the talk - Part I

Summary of the talk

Encryption

To protect confidentiality of data → use symmetric encryption.

- *Stream ciphers*.- Bits (or bytes) are enciphered/deciphered on-the-fly.
  - They offer the highest encryption speed.
  - They are transmission error-resilient.
  - Mainly used in telecommunication encryption, telephony encryption...
- *Block ciphers*.- Data are first split into blocks (usually 128-bit blocks).
  - Output blocks (plaintext, respectively ciphertext) are produced from both the same secret key and the input block (ciphertext, respectively plaintext).
  - They are not transmission error-resilient except in OFB mode.

Stream Ciphers

- A truly random (Vernam ciphers) or a pseudo-random sequence (finite-state cryptosystems) σ is bitwise combined to the text.
- The sequence σ is as long as the text

$$C_i = \sigma_i \oplus P_i$$

  where C_i, σ_i and P_i are the ciphertext, pseudo-random and plaintext sequences respectively.
- In Vernam ciphers, σ is produced by hardware methods. The key is duplicated before use.
- Any reuse of the key, even with a phase τ ($\sigma' = \sigma_{i+\tau}$) has a dramatic impact on the expected perfect secrecy (see white paper).

Stream Ciphers (2)

- For pseudo-random ciphers σ is produced by expanding a limited-size secret key by means of a finite-state algorithm

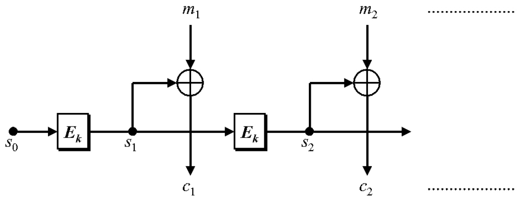
$$\sigma = E(K, K_P)$$

where K_P is a session or message key produced by the cryptosystem internals (message key generator module, software...).

- Strong requirement : the pair (K, K_P) must never be reused (derived from the Shannon's perfect secrecy).
- Most famous stream ciphers : E0 (*Bluetooth*), RC4, A5/1.
- Most stream ciphers are proprietary algorithms and thus are not public.

Block Ciphers

- The reuse of the key from block to block is supposed to have no impact on the overall security*.
- Block ciphers in *output feedback mode* (OFB) emulate stream ciphers.



- The secret key is the block s_0 and the pseudo-running sequence is made of blocks $s_1, s_2, s_3 \ldots$
- Block ciphers in OFB mode are fully equivalent to stream ciphers.

Problem Formalization

Definition

Two (or more) ciphertexts are said parallel *if they are produced from the same running key produced either by a stream cipher (Vernam cipher or finite state machine) or by a block cipher in OFB mode. If ciphertexts* $c_1, c_2 \ldots c_k$ *are parallel, the* parallelism depth *is* k.

- We have $C^1 = M^1 \oplus \sigma$ and $C^2 = M^2 \oplus \sigma$.
- Two issues to solve :
  1. **Detection issue**.- Among a huge number of ciphertexts, how to detect the different groups of parallel messages ?
  2. **Cryptanalysis issue**.- Once parallel messages have been detected, how to break the encryption and recover the plaintexts ?

Operational Requirements

- We do not care about the underlying cryptosystem (stream cipher or block cipher in OFB mode).
    - The system can remain totally unknown.
- Consequently we do not care about the secret key used either.
    - We do not need to perform a preliminary key recovery step.
    - $\Rightarrow$ *key-independent cryptanalysis*
- The cryptanalysis must be performed in polynomial time (e.g. within a reasonable amount of time).
- The parallelism depth must be at least equal to 2.

Summary of the talk

Weakness of Parallel Ciphertexts

- Let us consider two parallel ciphertexts $c_1 = c_1^0, c_1^1, c_1^2, c_1^3 \ldots$ and $c_2 = c_2^0, c_2^1, c_2^2, c_2^3 \ldots$.
- Since they are parallel, they are enciphered with the same (pseudo-)running sequence $\sigma = \sigma_0, \sigma_1, \sigma_2, \sigma_3 \ldots$ Let be $m_1 = m_1^0, m_1^1, m_1^2, m_1^3 \ldots$ and $m_2 = m_2^0, m_2^1, m_2^2, m_2^3 \ldots$ the corresponding plaintexts. We have

$$c_i^j = \sigma^j \oplus p_i^j \qquad \text{for all } i = 1, 2 \text{ and } j \leq N$$

where N is the size of the common parts of c_1 and c_2.

Weakness of Parallel Ciphertexts (2)

- Let us bitwise xor the two encrypted texts c_1 and c_2. Then we have :

$$c_1^j \oplus c_2^j = p_1^j \oplus \sigma^j \oplus p_2^j \oplus \sigma^j \qquad \text{for all } j \leq N$$

- Then, we have a quantity which no longer depends on the (pseudo-)running sequence :

$$c_1^j \oplus c_2^j = p_1^j \oplus p_2^j \qquad \text{for all } j \leq N$$

- Since it is the bitwise xor of two plaintexts, they have a very particular statistical profile.

Detecting Parallel Ciphertexts

- Under this assumption of parallelism, detecting parallel ciphertexts among a large amount of texts is very easy :
  - Equivalent to detect random files from non random files.
  - Very basic statistical testing.
- Bitwise xor every pair of texts and count Z the number of null bits in the resulting sequence. Then
  - If the two texts are not parallel then Z has a normal distribution law $\mathcal{N}(\frac{N}{2}, \frac{\sqrt{N}}{2})$.
  - Otherwise, Z has a has a normal distribution law $\mathcal{N}(np, \sqrt{p(1-p)})$ where $p > \frac{1}{2}$ is the probability for a bit to be zero (depends on the pair (language/encoding)).
- The test can explore thousands of text within a hour.
- To detect a complete set of parallel texts, just use the fact that parallelism is an equivalence binary relation.

Detecting Parallel ciphertexts (2)

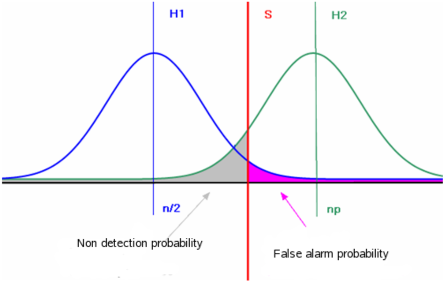
- Compute $Z = \sum_{i=1}^{N}(c_1^i \oplus c_2^i \oplus 1)$.
- Look for extremal values of Z.

z[1-2]	**6081**	**0.658**	z[3-15]	4677	0.506	z[6-18]	4611	0.499	z[10-20]	4629	0.501
z[1-3]	**6110**	**0.662**	z[3-16]	4604	0.498	z[6-19]	4586	0.496	z[11-12]	4608	0.499
z[1-4]	**6141**	**0.665**	z[3-17]	4692	0.508	z[6-20]	4660	0.504	z[11-13]	4670	0.506
z[1-5]	**6148**	**0.666**	z[3-18]	4606	0.499	z[7-8]	4647	0.503	z[11-14]	4582	0.496
z[1-6]	4695	0.508	z[3-19]	4605	0.499	z[7-9]	4657	0.504	z[11-15]	4573	0.495
z[1-7]	4636	0.502	z[3-20]	4627	0.501	z[7-10]	4580	0.496	z[11-16]	4582	0.496
z[1-8]	4607	0.499	**z[4-5]**	**6113**	**0.662**	z[7-11]	4594	0.497	z[11-17]	4584	0.496
z[1-9]	4545	0.492	z[4-6]	4634	0.502	z[7-12]	4668	0.505	z[11-18]	4642	0.503
z[1-10]	4638	0.502	z[4-7]	4585	0.496	z[7-13]	4626	0.501	z[11-19]	4591	0.497
z[1-11]	4652	0.504	z[4-8]	4626	0.501	z[7-14]	4550	0.493	z[11-20]	4593	0.497
z[1-12]	4560	0.494	z[4-9]	4622	0.500	z[7-15]	4667	0.505	z[12-13]	4548	0.492
z[1-13]	4682	0.507	z[4-10]	4621	0.500	z[7-16]	4548	0.492	z[12-14]	4592	0.497
z[1-14]	4634	0.502	z[4-11]	4703	0.509	z[7-17]	4642	0.503	z[12-15]	4591	0.497
z[1-15]	4653	0.504	z[4-12]	4627	0.501	z[7-18]	4562	0.494	z[12-16]	4614	0.500
z[1-16]	4578	0.496	z[4-13]	4629	0.501	z[7-19]	4625	0.501	z[12-17]	4574	0.495
z[1-17]	4642	0.503	z[4-14]	4565	0.494	z[7-20]	4629	0.501	z[12-18]	4508	0.488
z[1-18]	4606	0.499	z[4-15]	4664	0.505	z[8-9]	4514	0.489	z[12-19]	4549	0.492
z[1-19]	4601	0.498	z[4-16]	4611	0.499	z[8-10]	4531	0.491	z[12-20]	4619	0.500
z[1-20]	4639	0.502	z[4-17]	4655	0.504	z[8-11]	4617	0.500	z[13-14]	4598	0.498
z[2-3]	**6125**	**0.663**	z[4-18]	4537	0.491	z[8-12]	4661	0.505	z[13-15]	4677	0.506
z[2-4]	**6126**	**0.663**	z[4-19]	4590	0.497	z[8-13]	4589	0.497	z[13-16]	4646	0.503
z[2-5]	**6099**	**0.660**	z[4-20]	4592	0.497	z[8-14]	4709	0.510	z[13-17]	4622	0.500
z[2-6]	4590	0.497	z[5-6]	4625	0.501	z[8-15]	4530	0.490	z[13-18]	4648	0.503

- Here ciphertexts 1, 2, 3, 4 and 5 are parallel ("RC4-protected" Word files ; see later on).

Detecting Parallel Ciphertexts (3)

- Equivalent statistical test. Choose according to the value of Z with respect to a decision threshold S.



- S depends on the error probabilities you accept.
- This step is (plaintext) language/encoding independent !

Summary of the talk

Statistical Model of the Target Language

- Our aim : to recover the plaintexts directly without any preliminary step of key recovery.
- We need to build a qualitative and quantitative model of the target language.
- Language considered in the general sense (Chomsky taxonomy) : natural language, artificial languages (e.g. processor opcodes)...
- Never forget that in a computer context you must consider language AND encoding (ASCII, Unicode. . .) at the same time.

Statistical Model of the Target Language (2)

- Concept of corpus.
  - The set of all possible n-grams with their respective frequency of occurence.
  - Define a discrete random variable X describing any n-gram value. We denote p_i the probability that X takes the value x_i : $P(X = x_i) = p_i$ with $i \in \{0, 1 \ldots, N\}$.
  - N is the size of the corpus.
  - The distribution law of X is entirely determined by probabilities p_i of events $\{\{X = x_i\}\}$ where the x_i are the different n-grams in the corpus.

Statistical Model of the Target Language (3)

- First establish a n-grams corpus for the target language (set of n-grams with frequency).
- English is the easiest one to model.
- Optimal values are $n = 4$ or $n = 5$ ($n = 3$ works well if you have at least four parallel texts).

3-grammes	Fréquence	3-grammes	Fréquence
ENT	0,90	ELA	0,44
LES	0,80	RES	0,43
EDE	0,63	MEN	0,42
DES	0,61	ESE	0,42
QUE	0,60	DEL	0,40
AIT	0,54	ANT	0,40
LLE	0,51	TIO	0,38
SDE	0,51	PAR	0,36
ION	0,48	ESD	0,35
EME	0,47	TDE	0,35

- You can specialize your corpus (level of language, technical language...).
- A forensic and intelligence initial step may be useful.

Statistical Model of the Target Language (4)

- The n-grams corpus must be :
  - representative of the language level, context and nature used.
  - must be statistically admissible (compliant with Zipf law).
  - must describe a large enough character space.
- For most of the use, a 4-grams corpus built on modern language is sufficient.
- We have used a 96-character space

a	b	c	d	e	f	g	h	i	j	k	l	m
n	o	p	q	r	s	t	u	v	x	x	y	z
A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	.	,	;
:	?	!	«	(	)	{	}	+	-	*	/	=
‘	à	â	ç	è	é	ê	î	ô	ù	espace		

- Far easier for English texts.

Statistical Model of the Target Language (5)

- Language level and its impact on the corpus (qualitative aspect).

```
" de "  avec la fréquence 0.727323
" la "  avec la fréquence 0.405988
"ait "  avec la fréquence 0.405022
" et "  avec la fréquence 0.386859
"ent "  avec la fréquence 0.332413
" le "  avec la fréquence 0.323777
"les "  avec la fréquence 0.315910
"tion"  avec la fréquence 0.296196
"e de"  avec la fréquence 0.264861
"que "  avec la fréquence 0.259306
```

```
" de "  avec la fréquence 0.637018
" la "  avec la fréquence 0.348194
"ent "  avec la fréquence 0.332461
"ait "  avec la fréquence 0.318509
"les "  avec la fréquence 0.303970
" et "  avec la fréquence 0.294795
"e de"  avec la fréquence 0.274558
" le "  avec la fréquence 0.266932
"que "  avec la fréquence 0.251702
" les"  avec la fréquence 0.251325
```

```
" de "  avec la fréquence 0.895318
"tion"  avec la fréquence 0.784308
"atio"  avec la fréquence 0.486838
"ion "  avec la fréquence 0.471748
" la "  avec la fréquence 0.466020
"ent "  avec la fréquence 0.437264
"ment"  avec la fréquence 0.414586
"les "  avec la fréquence 0.396094
"des "  avec la fréquence 0.390308
" des"  avec la fréquence 0.382370
```

FIGURE: Corpus built respectively on non-modern (left), modern (center) and modern military texts (right).

- Use of hash table to limit memory/time ressources.

Cryptanalysis Principle

- Let be $C_1, C_2 \ldots, C_i \ldots, C_p$ p parallel ciphertexts to decrypt.
- Consider a corpus of N n-grams (typically $n = 4$) and let us denote those n-grams by $x_0 \ldots x_N$.
- Split the ciphertexts into a succession of n-grams.
- Decryption algorithms main steps are :
  1. for each ciphertext n-gram C_1^j in the first ciphertext C_1, make an assumption on the corresponding plaintext n-gram denoted M_1^j. This n-gram M_1^j is exhaustively searched through the set $\{x_0 \ldots, x_N\}$ of n-grams in the working corpus ;
  2. compute the resulting key n-gram as follows : $K_j = C_1^j \oplus M_1^j$;
  3. apply K_j to each of the corresponding ciphertext n-grams in the $(p-1)$ remaining ciphertexts : $M_i^j = C_i^j \oplus K_j, \ i \in \{2, p\}$;
  4. repeat the previous steps exhaustively for every n-grams in the corpus.

Cryptanalysis Principle (2)

- The algorithm computes N p-tuples $(M_1^j, M_2^j \ldots, M_p^j)$ for each ciphertext n-gram at index j.
- Each such p-tuple represents plaintext n-gram candidates for plaintext messages $(M_1, M_2 \ldots, M_p)$ at index j.
- To determine which is the most probable one, associate to each of the N p-tuples of n-grams, the corresponding p-tuple of probabilities $(P[M_1^j], P[M_2^j] \ldots, P[M_p^j])$.
- The most probable plaintext n-grams p-tuple is the one which maximizes the p-tuples of probabilities.

Cryptanalysis Principle (3)

- The issue is to choose a suitable function to process those probabilities in the most significant way :

$$Z_j = f(P[M_1^j], P[M_2^j] \dots, P[M_p^j])$$

- The choice of this function strongly depends on the nature of the texts (presence of a many proper or geographical names, technical terms. . .).
- The function must always be a strictly increasing positive function.
- Strong impact of the skills and the experience of the cryptanalyst.
- The cryptanalysis algorithm has a polynomial complexity in $\mathcal{O}(pM)$ where M is the size (in bytes) of the ciphertexts.

General Algorithm

Input: p parallel ciphertexts $C_1, C_2 \ldots C_p$
Input: A N n-grams corpus $\{x_0, x_1 \ldots x_N\}$ of respective probabilities $\{P[x_0], P[x_1] \ldots P[x_N]\}$.
Output: p plaintexts $M_1, M_2 \ldots M_p$.

```
for all ciphertext n-gram C_1^j at index j in C_1 do
    Z_j = 0
    for all m_1^j ∈ {x_0, x_1 ... x_N} assume that M_1^j = m_1^j do
        Compute K_j = C_1^j ⊕ m_1^j
        For i ∈ {2, ..., p} do
            Compute m_i^j = C_i^j ⊕ K_j
            Store P[m_i^j]
        End For
        If f(P[m_1^j], P[m_2^j] ..., P[m_p^j]) > Z_j Then
            Z_j = f(P[m_1^j], P[m_2^j] ..., P[m_p^j])
            For i ∈ {2, ..., p} do
                M_1^j = m_1^j
            End For
        End If
    End For
End For
```

Basic Illustrative Example

C_1	t	3	X	;			t	3	X	;	
T_1	**A**	**r**	**m**	**y**	f_1		D	p	q	i	f'_1
K	0x35	0x41	0x35	0x42			0x30	0x43	0x29	0x52	
C_2	f	$	V	0			f	$	V	0	
K	0x35	0x41	0x35	0x42			0x30	0x43	0x29	0x52	
T_2	**S**	**e**	**c**	**r**	f_2		V	9	?	b	f'_2
C_3	{	4	˜	'			{	4	˜	'	
K	0x35	0x41	0x35	0x42			0x30	0x43	0x29	0x52	
T_2	**N**	**u**	**K**	**e**	f_3		K	w	W	u	f'_3

FIGURE: Correct (left) and wrong plaintext guess (? means non printable)

- Obviously $F(f_1, f_2, f_3) > F(f'_1, f'_2, f'_3) \Rightarrow$ correct guess at left.

Key Parameters

- A few parameters have a significant impact on the final probability of success :
  - the frequency function F,
  - the decrypting mode,
  - the decision mode.
- A number of refinements enable to drastically speed up the cryptanalysis and increase the final probability of success to recover the whole texts.

Frequency Function F

- It must be a strictly positive increasing function.
  - Either additive

$$F(f_1, f_2, \ldots, f_k) = \sum_{i=1}^{k} f_i^a$$

  - Or multiplicative

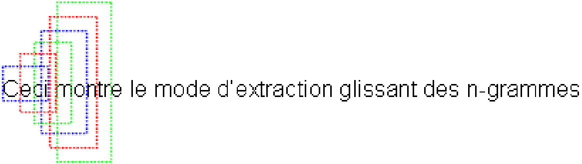
$$F(f_1, f_2, \ldots, f_k) = \prod_{i=1}^{k} (f_i^a + 1)$$

- The multiplicative one is far more efficient since it amplifies the impact of frequent n-grams while limiting the effect of marginal frequencies of rare (but correct) plaintext n-grams.
- The value $a = 0.3$ is optimal.

Decrypting Mode

- It depends on the way n-grams are taken in the ciphertexts.
  - Either normal mode : n-grams have void intersection (consecutive). This mode is the less efficient one.

  - Or overlapping mode : n-grams share $(n-1)$ characters.



- The overlapping mode allows a large number of optimizations and algorithmic tricks. It is therefore the most efficient.
- The non empty intersection enables to greatly increase the confidence in the final plaintext n-gram we keep.

Decrypting Mode : Basic Example

S	**W**	**E**	**E**							
	W	H	**E**	R						
		E	R	E	S					
			E	**T**	**N**	**I**				
				T	O	N	I			
					N	**I**	**G**	E		
						I	**G**	**H**	B	
							G	**H**	O	S
								H	**T**	A
									T	**I**
										I
S	W	E	E	T	N	I	G	H	T	I

- Somehow a mix of maximum-likelyhood decoding (quantitative aspect) and coherence decoding (qualitative aspect).
- Optimize the decrypting success at the end of the texts (common part).

Decision Mode

- This cryptanalysis consists somehow in performing a decoding. It is then possible to use ECC techniques.
- Either hard decision : for every n-gram index, we keep only the best candidate.
    - Any n-gram error will be difficult to recover and the final plaintext may contain a significant number of "holes".
    - Problematic when the plaintext contains rare n-grams (proper name, technical terms...).
- Or soft decision : for every n-gram index, we keep up to the p best candidates.
- Can prevent a bad decision at previous index (e.g. the correct n-gram has the second best score).
- A little bit more tricky to implement but far more efficient.

Refinements and Optimization

- The best approach consists in combining all the previous key elements.
  - multiplicative frequency function F with $a = 0.3$,
  - overlapping mode with all optimizations enabled,
  - soft decision $(5 \leq p \leq 10)$.
- It is however possible to increase the efficiency of the cryptanalysis by considering a few other refinements.
- Reject guesses which produce n-grams containing characters that are not in the character space chosen (e.g. non printable character).

Refinements and Optimization (2)

- Performs semantic analysis on-the-fly of the m plaintext candidates when guessing a new n-grams (see language as Markov process).
  - It is necessary when having only two parallel ciphertexts.
  - There is an additional degree of freedom to deal with :

THER	EISA	ROTA	TING	EFFE	CT,
WHEN	DEAL	INGW	ITHT	WOTE	XTS

and

THER	DEAL	ROTA	TING	WOTE	XTS
WHEN	EISA	INGW	ITHT	EFFE	CT,

are statistically identical solutions but semantically different.
  - Semantic step has a local effect only. Can be combined by considering languages as Markov process (e.g. French language is a 19-Markov process).

Summary of the talk

Microsoft Office Market

- Microsoft Office represents
  - 75 % of office suites for home use.
  - 80 % of office suites for professional use.
- Most of the versions in use are Office versions up 2003 releases (version 11).
- Office still represents a small part of the market.
  - Microsoft Office 2007 version failed to attract many users because of a disconcerting break of ergonomics and a lack of easy-to-use features.

Microsoft Office Encryption

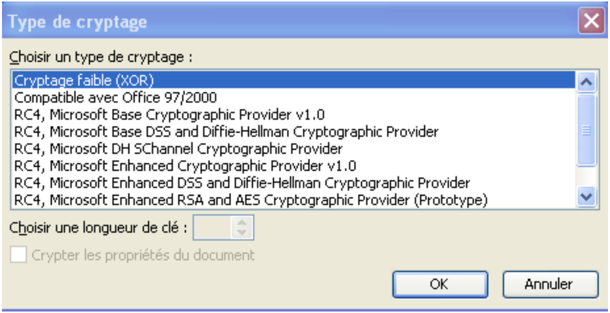
- Office provides password-based document encryption for every application of the suite.
- Different levels of encryption available sometimes.
- The default level is weak lame XOR encryption.
- What about the so-called most secure levels ?
  - Use of 128-bit key RC4 (up to Office 2003).
  - Really strong ?
- What the impact of the Windows operating system on the overall cryptographic security ?
- Let us broaden the debate : how to hide a decrypting trapdoor ?
- Without loss of generality, we focus on the Word application.

Our results

- Based on theoretical works of Hongju Wu (2004) (have never been practically exploited).
- We manage to decrypt operationally any Office documents protected with embedded encryption.
  - Any security level, including 128-bit key RC4, up to Office 2003.
- The practical attack relies both on cryptographic and forensic techniques that must be combined.
- Ideal combination for forensics purpose that can be envisaged as a trapdoor.
- The cryptanalysis can be performed within a couple of minutes.

Password-based Protection

- Usually through the *Tools* → *Options* menu.
  - Use the *Security* → *Advanced* tab.
- Different level of cryptographic security : from lame to supposedly high level.



XOR Encryption

- It is the default setting unless you use the *Advanced* tab.
  - Essentially to ensure the backward compatibility with former Microsoft Office suites.
- It is the lamest encryption method ever.
  - Mask the text with a constant pattern.

Plaintext	T	E	X	T	_	E	X	E	M	P	L	E
					$\oplus$							
Key	A	B	C	D	A	B	C	D	A	B	C	D
					$=$							
Ciphertext(hex)	15	7	$1B$	10	61	7	$1B$	1	C	12	1	1

  - Easy to detect (basic statistical test).
  - Easier to break.

XOR Encryption (2)

- Very characteristic to detect.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000A00	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»ÔB ý³ý Þ® ±▌î
00000A10	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»ÔB ý³ý Þ® ±▌î
00000A20	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»ÔB ý³ý Þ® ±▌î
00000A30	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»ÔB ý³ý Þ® ±▌î
00000A40	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»ÔB ý³ý Þ® ±▌î
00000A50	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»ÔB ý³ý Þ® ±▌î

- Very weak key management.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
000001F0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000200	EC	A5	C1	00	71	60	09	04	00	00	F0	93	BF	00	EB	CB	ì¥Á.q`....ð▌¿.ëË
00000210	C6	1F	00	30	00	00	00	00	00	06	00	00	06	08	00	00	Æ..0............

- The 32-bit hash of the password is stored at offset 0x20E.
- Immediate to break with dedicated software.
- Easy to break with classical cryptanalysis techniques.

RC4 Encryption

- All other Office encryption methods are using RC4.
- RC4 is a 2048-bit key stream cipher.
  - The key is limited to 40 bits in Office 97/Office 2000.
  - The key is extended to 128 bits in later Office suites (up to Office 2003).
- A pseudo-random sequence σ is expanded by RC4 from the key and combined to the text.

RC4 Encryption (2)

- The application builds the key K from the user password :

$$K = F(H(\text{IV}||\text{password}))$$

  where F is a 128-bit derivation function, H is a hash function (SHA-1) and IV is a 128-bit random initialization vector.
- The IV is located after the **10 00 00 00** marker (offset 0x147C).

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00001420	01	00	00	00	30	06	3B	02	00	00	00	00	4D	00	69	00	....0.;.....M.i.
00001430	63	00	72	00	6F	00	73	00	6F	00	66	00	74	00	20	00	c.r.o.s.o.f.t. .
00001440	53	00	74	00	72	00	6F	00	6E	00	67	00	20	00	43	00	S.t.r.o.n.g. .C.
00001450	72	00	79	00	70	00	74	00	6F	00	67	00	72	00	61	00	r.y.p.t.o.g.r.a.
00001460	70	00	68	00	69	00	63	00	20	00	50	00	72	00	6F	00	p.h.i.c. .P.r.o.
00001470	76	00	69	00	64	00	65	00	72	00	00	00	10	00	00	00	v.i.d.e.r.......
00001480	ED	F9	CE	9B	DA	F1	80	0F	F2	AC	65	2C	57	44	62	1D	íùÎ▌ÚÑ▌.ò¬e,WDb.
00001490	4C	FD	1F	DE	21	AF	A6	50	91	A3	47	2C	E5	22	DD	BA	Lý.Þ!¯¦P'£G,å"Ýº

RC4 Encryption (3)

- This encryption is supposed to be secure provided that :
  - The sequence is unique for every different document (even up to one byte).
  - The key does not depend on the password only.
  - The key space is large enough.
- In this respect, RC4-based Office encryption seems to be secure.
- In fact, this encryption is weak and can be operationally broken (see further).

Word Document Critical Fields

- To conduct the cryptanalysis, it is necessary to identify a few internals of Office documents (e.g. Word here).
  - We need to know where the text begins and its size (in other words where it ends).
  - Text has variable length by nature.
- The text (encrypted or not) always begins at offset 0xA00.
- To calculate the text length, look at offsets 0x21C and 0x21D. Let be x and y the values respectively found here.
  - The text length L is then given by

$$L = (y - 8) \times 2^8 + x$$

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000200	EC	A5	C1	00	71	60	09	04	00	00	F0	13	BF	00	B8	00	ì¥Á.q`....ð.¿.¸.
00000210	00	00	00	30	00	00	00	00	00	06	00	00	$\boxed{7E}$	$\boxed{08}$	00	00	...0........~...
00000220	0E	00	62	6A	62	6A	71	50	71	50	00	00	00	00	00	00	..bjbjqPqP......

Office Encryption Vulnerability

- Theoretically identified by Hongju Wu in 2004. Never verified on an practical/operational basis.
- Based on the fact that Office uses the same IV for every different versions (revisions) of a given document.
  - The user generally does not change the password from revision to revision. So the key K remains the same.
  - This flaw cannot be exploited with a single text. A revision is supposed to overwrite the previous one.
  - No so obvious to implement a cryptanalysis using it.
  - It supposes also a weakness at the operating system level.
- Interesting issue : can we consider the combination of two (suitable) flaws as a trapdoor ?

Highlighting the Flaw

- We slightly modify a *Word* document (one-word insertion ; e.g. changing the date).
  - Original text : "*Ceci est un essai de construction de messages parallèles afin de montrer la vulnérabilité du chiffrement de Microsoft Word*".

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000A00	31	37	B2	B6	3E	AD	B6	45	F4	B5	B9	0F	D3	25	44	33	17²¶>–¶Eôµ¹.Ó%D3
00000A10	09	21	DA	67	BC	DC	07	2F	62	13	A7	F6	0F	1D	D8	FC	.!Úg¼Ü./b.§ö..Øü
00000A20	51	07	DA	C6	98	77	C4	CC	FC	3B	54	D7	1E	38	C4	1C	Q.ÚÆ▮wÄÌü;T×.8Ä.
00000A30	E1	02	E5	BC	96	16	98	55	3B	4F	D3	0E	7E	97	86	B0	á.å¼▮.▮U;OÓ.~▮▮°
00000A40	C0	73	F9	67	24	60	12	7C	2B	60	A6	F1	F2	76	A4	E6	Àsùg$`.|+`¦ñòv¤æ
00000A50	03	AC	F5	1F	14	0A	A6	84	7D	8B	0C	E4	2D	B5	A0	75	.¬õ...¦▮}▮.ä-µ u
00000A60	2C	28	F5	[illegible]	E6	61	27	8D	F6	18	D2	33	DC	7C	04	A6	,(õ¾æa'▮ö.Ò3Ü|.¦
00000A70	C7	A4	37	B3	2E	D1	6B	D5	DE	93	58	59	1C	90	E6	09	Ç¤7³.ÑkÕÞ▮XY.▮æ.

  - Modified text : "*Ceci est un essai de construction de* **deux** *messages parallèles afin de montrer la vulnérabilité du chiffrement de Microsoft Word*".

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000A00	31	37	B2	B6	3E	AD	B6	45	F4	B5	B9	0F	D3	25	44	33	17²¶>–¶Eôµ¹.Ó%D3
00000A10	09	21	DA	67	BC	DC	07	2F	62	13	A7	F6	0F	1D	D8	FC	.!Úg¼Ü./b.§ö..Øü
00000A20	51	07	DA	C6	98	7E	C4	CA	F7	7A	5E	D7	1E	6B	D5	1A	Q.ÚÆ▮~ÄÊ÷z^×.kÕ.
00000A30	F6	10	A9	A0	1F	08	9C	4A	77	C6	D9	02	63	97	83	B3	ö.© ..▮JwÆÙ.c▮▮³
00000A40	89	70	B6	6D	35	32	1A	61	65	78	B5	B4	F6	23	A4	E9	▮p¶m52.aexµ´ö#¤é
00000A50	CA	A8	E1	11	13	8F	BD	91	F6	C2	04	F8	79	3F	E8	78	Ê¨á..▮½'öÂ.øy?èx
00000A60	3F	6E	E4	B3	E2	62	2F	8B	B3	11	D2	7D	E5	35	03	B1	?nä³âb/▮³.Ò}å5.±
00000A70	88	9A	31	B6	28	9E	4F	D5	CA	83	56	03	73	E2	82	27	▮▮1¶(▮OÕÊ▮V.sâ▮'

Exploiting Another Weakness

- The main problem lies in the fact that normally each new version of a text should overwrite the previous one.
- Then in an ideal operating system, the parallism depth (number of parallel encrypted documents) should be equal to 1.
- The cryptanalysis is therefore not possible.
- Perfection lies elsewhere.
  - There is another weakness in Windows system which looks innocent in itself : temporary files + unsecure erasing.
  - It is then possible to increase the parallelism depth (sometimes in a very important way).
- Combining the two gives a powerful ability for any forensic analysis.

Increasing Parallelism Depth

- Temporary files (one per revision !).

- They are unsecurely deleted : use a recovery software !

Contenu de 'Effacé(s)\rapport_confidentiel'

Nom	Taille	Date de modific...	MFT entry	Condition	Type
~$nfidentiel_chiffre.doc	162	15.08.2008 11:05	47002	good	Document Micro...
~WRL0004.tmp	50176	15.08.2008 11:05	47377	good	Fichier TMP
~WRL2361.tmp	50176	15.08.2008 11:06	47383	good	Fichier TMP
~WRL4027.tmp	50176	15.08.2008 11:05	46262	good	Fichier TMP

- In average, the parallelism depth is about 4 to 6.
- It is very easy to steal all these versions with a simple (malicious) USB key. It then goes beyond simple forensic aspects.

Experimental Results

- We have performed a lot of experiments on different languages (from different linguistic groups).
  - Test group 1 : Common language/non modern texts.
  - Test group 2 : Common language/modern texts.
  - Test group 1 : Technical language/modern texts.

Nombre de textes parallèles	Nombre de caractères correctement décryptés			Pourcentage de bon décryptement		
	Test1	Test2	Test3	Test1	Test2	Test3
2 parallèles	462	633	3845	40,07 %	40,66 %	39,80 %
3 parallèles	1018	1283	8679	88,29 %	82,40 %	89,78 %
4 parallèles	1069	1414	8880	92,71 %	90,81 %	91,87 %
5 parallèles	1081	1428	9001	93,76 %	91,71 %	93,12 %

- With full optimization enabled, the probability of success if very close to 100 %.
- Just require a final check by human operator to manage proper names or very rare terms.

Summary of the talk

The Excel Case

- This case is less easy to solve but the principle remains the same. We manage to recover data from parallel texts as efficiently as for Word.
  - The offset of data beginning is variable.
  - The data structure are quite different (cells instead of text).
  - The nature of data are different (numbers rather than letters).
  - Modifications of cells are stored at the end of the sheet data.
- But to bypass these problems, we observed and use the fact that
  - Data are always beginning 31 bytes after the 0x8C000400 pattern.
  - The end marker depends on the number of cells in the sheet. Data are ending right before the 0xFF001200 $+ \alpha$ pattern where

    $$\alpha = (8 \times p) \times 256$$

    Hence we have this marker equal to 0xFF000a00, 0xFF001200, 0xFF1a00. . . .

Excel Modifications

Let us consider a text and its revision.

	A	B	C
1			
2		colonne 1	colonne 2
3	ligne 1	données 11	données 12
4	ligne 2	données 21	données 22

	A	B	C
1			
2		colonne 1	colonne 2
3	ligne 1	données 11	modification
4	ligne 2	données 21	données 22

Viewing modifications

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
000C08F0	FC	00	6A	00	08	00	00	00	08	00	C0	00	0A	00	00	63	ü j c
000C0900	6F	6C	6F	6E	6E	65	20	31	20	09	C0	00	63	6F	6C	6F	olonne 1 colo
000C0910	6E	6E	65	20	32	08	00	00	6C	69	67	6E	65	20	31	20	nne 2 ligne 1
000C0920	07	00	00	6C	69	67	6E	65	20	32	CA	00	00	64	6F	6E	ligne 2 don
000C0930	6E	E9	65	73	20	31	31	0A	00	00	64	6F	6E	6E	E9	65	nées 11 donnée
000C0940	73	20	32	31	0A	00	00	64	6F	6E	6E	E9	65	73	20	31	s 21 données 1
000C0950	32	0A	00	00	64	6F	6E	6E	E9	65	73	20	32	32	FF	00	2 données 22ÿ

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000910	6C	00	08	00	00	03	08	00	00	00	0A	00	00	63	6F	6C	l col
00000920	6F	6E	6E	65	20	31	20	09	00	00	63	6F	6C	6F	6E	6E	onne 1 colonn
00000930	65	20	32	08	00	03	6C	69	67	6E	65	20	31	20	07	00	e 2 ligne 1
00000940	00	6C	69	67	6E	65	20	32	0A	00	00	64	6F	6E	6E	E9	ligne 2 donné
00000950	65	73	20	31	31	0A	00	00	64	6F	6E	6E	E9	65	73	20	es 11 données
00000960	32	31	0A	00	00	64	6F	6E	6E	E9	65	73	20	32	32	0C	21 données 22
00000970	00	00	6D	6F	64	63	66	69	63	61	74	69	6F	6E	FF	00	modificationÿ

The Encryption Flaw in Excel

- Let us consider an encrypted text and its encrypted revision.

	A	B	C
1	luke	obi wan	yoda
2	yan solo	leia	chewbacca

	A	B	C
1	luke	obi wan	yoda
2	yan solo	princesse leia	chewbacca

- Identifying the flaw.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00C009B0	D9	2D	C8	41	51	16	A6	E8	8C	00	04	00	30	18	19	AC
00C009C0	C1	01	08	00	CF	19	A9	81	03	D3	33	3C	FC	00	3E	00
00C009D0	E6	0D	D3	70	E7	27	29	8B	A7	C0	FE	06	7E	83	59	95
00C009E0	3D	35	F3	46	4D	91	3E	3D	C2	9C	81	4F	AD	E3	30	A8
00C009F0	29	44	1C	18	CA	B7	81	0B	18	5C	62	DB	64	DE	D1	67
00C00A00	DD	7F	DE	24	CB	C3	1D	2E	66	8B	4D	4F	5C	09	FF	00
00C00A10	0A	00	20	F0	9D	D0	2C	FD	0E	3F	18	95	3A	00	00	00

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000009B0	D9	2D	C8	41	51	16	A6	E8	8C	00	04	00	30	18	19	AC
000009C0	C1	01	08	00	CF	19	A9	81	03	D3	33	3C	FC	00	48	00
000009D0	E6	0D	D3	70	E7	27	29	8B	A7	C0	FE	06	7E	83	59	95
000009E0	3D	35	F3	46	4D	91	3E	3D	C2	9C	81	4F	AD	E3	30	A8
000009F0	29	44	1C	18	CA	B7	81	0B	18	5C	62	D6	64	DE	DE	6A
00000A00	D1	69	B5	45	A8	C3	14	45	11	E9	5C	5E	66	06	DC	56
00000A10	9C	60	4D	D0	29	B2	45	9C	FF	00	0A	00	41	4D	C4	6F

Detecting Excel Parallel Files

- The principle remains exactly the same.

```
z[1-2] 1728 0.651584
z[1-3] 1372 0.500730
z[1-4] 1347 0.511002
z[1-5] 1091 0.507914
z[1-6] 952 0.501053
z[2-3] 1358 0.512066
z[2-4] 1332 0.505311
z[2-5] 1028 0.478585
z[2-6] 974 0.512632
z[3-4] 1322 0.501517
z[3-5] 1083 0.504190
z[3-6] 947 0.498421
z[4-5] 1048 0.487896
z[4-6] 927 0.487895
z[5-6] 929 0.488947
```

- No significant difference with Word.

Excel Cryptanalysis

- The principle remains exactly the same as well.
- Two additional constraints however to deal with.
  - Data include specific (cell) separator fields of the form `XX 00 00`

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
000008F0	FC	00	6A	00	08	00	00	00	08	00	00	00	0A	00	00	63	ü j c
00000900	6F	6C	6F	6E	6E	65	20	32	20	0A	00	00	63	6F	6C	6F	olonne 2 colo
00000910	6E	6E	65	20	31	20	07	00	00	6C	69	67	6E	65	20	31	nne 1 ligne 1
00000920	07	00	00	6C	69	67	6E	65	20	32	0A	00	00	64	6F	6E	ligne 2 don
00000930	6E	E9	65	73	20	31	31	0A	00	00	64	6F	6E	6E	E9	65	nées 11 donnée
00000940	73	20	31	32	0A	00	00	64	6F	6E	6E	E9	65	73	20	32	s 12 données 2
00000950	32	0A	00	00	64	6F	6E	6E	E9	65	73	20	32	31	FF	00	2 données 21ÿ

    In fact this constraint turns to be a very interesting feature since it is very probable plaintext AND it enables to recover from wrong n-gram guesses regularly.
  - Use a specific n-gram corpus (no sentences, different space character, very few verbs, mainly numbers...).
- The parallelism depth is generally higher than for Word.
- Decrypting Excel proved to be efficient and operationally feasible.

Summary of the talk

Work Summary

- We have designed a fully operational technique/tools to detect and break any misused or badly implemented stream ciphers and block ciphers (OFB mode).
  - Mainly concern forensics needs.
  - Also address cryptanalysis of unknown encrypted communications.
  - However applicable through an attack to steal the parallel texts (malicious USB key, spy malware...).
- Existing cases more numerous than expected and/or suspected.
- No knowledge required about the cryptosystem.
- No time-consuming key recovery step required.

Cryptology Issues

- When cryptography works on paper, the real security can be very still very far.
- The implementation can be (intentionally or not) flawed.
- Critical modules (e.g. message key generator) may fail.
- Our method enables to detect these cases without performing time-consuming, complex reverse-engineering steps.

What about Cryptographic Trapdoors ?

- What is a flaw can be in reality an (intended) trapdoor when combined to another flaw.
- Especially when the two flaws are maintained thoughout time and versions (of Office AND Windows).
- Give a very interesting insight on how to build such trapdoors.
  - Just use more than two innocent-looking flaws (50 % at the application level, 50 % at the OS level).
  - Exploit the fact that misuses will occur with a very high probability.
  - Use secret-sharing schemes or threshold schemes.
- The choice of the encoding is also part of the game (CCITTx vs ASCII).

What about Cryptographic Trapdoors ? (2)

- Can be interestingly extended to cryptosystems themselves (e.g. block ciphers) to produce trapped encryption.
  - Design a commercial encryption software labelled "AES-256 inside".
  - Implement it in OFB mode with IVs produced at the OS level.
  - Introduce a flaw at the OS random number generator level.
  - Use a malware to exploit this flaw in such a way that fixed IVs are produced.
- Guess what is the result ?
- Many other scenarii possible. Just let play your imagination.

Questions

- Many thanks for your attention.
- Questions ... (there is no stupid questions !)...
- and Answers ...(there are eventually just stupid answers).

Now let us go to practice and real cases (Tutorial Part II)...