Word と Excel の暗号を分析する

Analyzing Word and Excel Encryption

An operational solution

Eric Filiol
ESIEA - Laval
Operational Cryptology and Virology Lab
$(C + V)^O$
Pacsec 2009

Microsoft Office の暗号

- Office の各アプリケーションは、パスワードを用いた文書の暗号化を行っている.
- 異なるレベルでの暗号化が可能な場合もある.
- “標準レベル” での暗号化には xor 暗号を用いているが、安全とは呼べない.
- “安全な(と表記された)レベル” ではどうか？
  - 鍵長 128-bit の RC4 を用いている (Office 2003 まで).
  - 本当に安全か？
- 全ての暗号化セキュリティに対する Windows OS の影響は？
- 議論を広げよう: 復号トラップをどうやって秘匿するのか？
- Word に限定して話を進める (それでも問題は生じない)

Microsoft Office とマーケット

- Microsoft Office は
  - 家庭利用のオフィススイーツの 90 % を占める.
  - 仕事利用のオフィススイーツの 80 % を占める.
- 現在使用されている Office のほとんどは 2003 年までに発売されたもの (version 11).
- マーケットにおいて Office の占める割合は小さい.
  - 企業やユーザは Office 2007 への移行に消極的.
  - 互換性や操作の不慣れの問題.

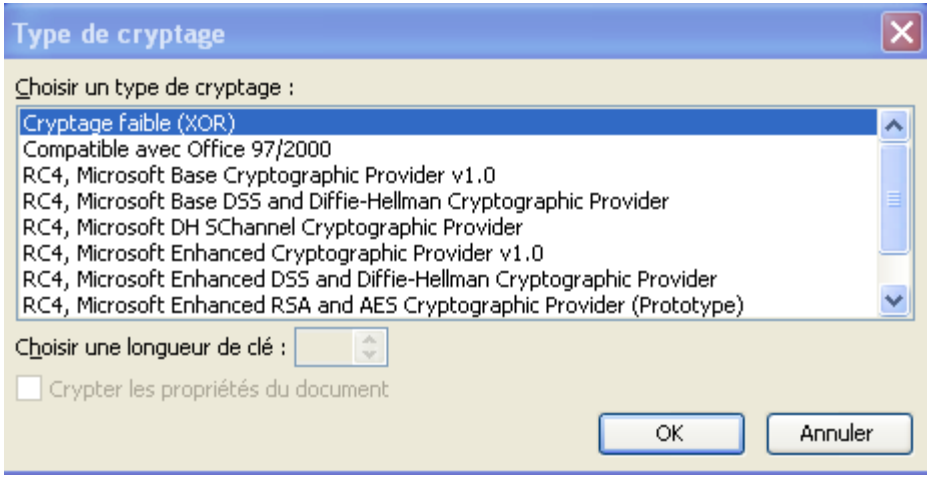
成果

- 2004年の Hongju Wu による理論的な分析結果を用いた(この分析結果がこれまでに検証されたことはなかった)
- 暗号化によって保護された Office の任意の文書の復号操作に成功した.
  - 鍵長 128-bit の RC4 を含む全てのセキュリティレベル
  - ただし Office 2003 まで
- 実際の攻撃には、暗号学的な技術とフォレンジック技術を組み合わせる必要がある
- トラップとも考えられるフォレンジック用途には理想的
- 解読は数分で実行可能
- 実装には(Franck Bonnard の助けを借りて)C 言語を用いた.

目次

パスワードによる保護

- 通常はメニューの [ツール] → [オプション] で設定.
  - [セキュリティ] タブの [詳細設定] を選択
- さまざまなセキュリティレベルの暗号が利用可能: 安全でないものから(おそらくは)非常に安全なものまで



xor 暗号

- (“詳細設定” で設定されていないときに) 標準で使用される暗号方式
  - 以前の Microsoft Office アプリケーションの後方互換性を保証することが主な目的
- 最も弱い暗号方式
  - テキストを固定パターンでマスク

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

  - (基本的な統計テストにより) 検知するのは簡単
  - 解読はもっと簡単

xor 暗号（２）

- 特徴的なため検知しやすい

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\»Ôß ý³ý Þ® ±…î
00000A10	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»Ôß ý³ý Þ® ±…î
00000A20	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»Ôß ý³ý Þ® ±…î
00000A30	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»Ôß ý³ý Þ® ±…î
00000A40	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»Ôß ý³ý Þ® ±…î
00000A50	38	5C	BB	D4	DF	11	FD	B3	FD	11	DE	AE	02	B1	85	EE	8\»Ôß ý³ý Þ® ±…î

- 鍵管理がおろそか

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..........

- パスワードのハッシュ値(32-bit)がオフセット 0x20E に格納されている.
- 解読用ソフトを用いれば一瞬で解読できる.
- 古典的な暗号解析技法を用いても簡単に解読できる.

RC4 暗号

- Office の他の暗号化方式は RC4 暗号を使用
- RC4は最大鍵長が2048ビットのストリーム暗号
  - Office 97/2000 では鍵長が40ビットに制限されている
  - それ以降（Office 2003 まで）は鍵長が128ビットに拡張されている
- RC4 暗号によって鍵から擬似乱数系列σが生成され、テキストに重ねあわされる
- 擬似乱数系列 $C_i = \sigma_i \oplus P_i$ と同じ長さだけ生成される

ここで、C_i, σ_i, P_i はそれぞれ暗号文、擬似乱数、平文の系列である

RC4 暗号（２）

- アプリケーションはユーザパスワードから鍵 *K* を生成:

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

  ここで *F* は導出関数（出力長128ビット), *H* はハッシュ関数 (SHA-1、
  出力長160ビット), IVは乱数の初期ベクトル（128ビット）
- IV は「**10 00 00 00**」マーカーの後に記録されている (オフセット 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"Ý²

RC4 暗号（３）

- この暗号は以下のようになっていれば安全と考えられる:
  - 異なるドキュメントには（1バイトだけでも）異なる疑似乱数系列を使用
  - 同じパスワードでも異なる鍵が生成できる
  - 鍵空間（鍵の取りうる値の空間）は十分大きい
- この観点では、RC4 暗号を用いた Office の暗号は安全であるように見える
- しかし、実際にはこの暗号は脆弱であり、本当に解読することができる（この後に注目...）

Word文書の重要なフィールド

- 暗号解析を行うためには、オフィス文書（ここではWord文書）内のいくつかの内部情報を特定する必要がある
  - テキストの開始位置とそのサイズ（つまり終了位置）
  - テキスト長は可変であるのが普通
- 暗号化の有無に関わらず、テキストは常にオフセット0xA00から始まる
- テキスト長を求めるには、オフセット 0x21C と 0x21D を調べれば良い。ここに格納されている値を x, y とする。
  - このときテキスト長 L は次の式によって求められる

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

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
00000210	00	00	00	30	00	00	00	00	00	06	00	00	7E	08	00	00
00000220	0E	00	62	6A	62	6A	71	50	71	50	00	00	00	00	00	00

Office の暗号の脆弱性

- 2004年に Hongju Wu により理論的に発見される。しかし、現実的、運用的に検証されていなかった
- この脆弱性はバージョン(リビジョン)にかかわらず、Office は同じ IV をドキュメントに使用するという事実に基づいている
  - ユーザは リビジョンが変わってもパスワードを変えないのが普通なので、鍵 K は同じままである
  - この不具合は1つの文書では悪用できない. 更新された文書は以前の文書を上書きすると仮定している
  - この脆弱性を利用して暗号解析するのは簡単でない
  - さらにOSレベルの脆弱性も想定する
- 興味深い課題: 2つの(適切な)脆弱性を組み合わせることによって暗号解析することは可能か?
- 同じ暗号化操作が施された2つ(以上)の文書を「パラレル暗号化テキスト」と呼ぶこととする.

パラレル (暗号化) テキストの問題点

- 2つのパラレル暗号化テキスト c_1, c_2 を考える

  $c_1 = c_0^1, c_1^1, c_2^1, c_3^1, \ldots$

  $c_2 = c_0^2, c_1^2, c_2^2, c_3^2. \ldots$

- これらはパラレルなので、同じ擬似乱数系列

  $\sigma = \sigma_0, \sigma_1, \sigma_2, \sigma_3$

  を用いて暗号化されている (RC4による鍵 K の拡大)

- 対応する平文を p_1, p_2 とする

  $p_1 = p_0^1, p_1^1, p_2^1, p_3^1, \ldots$

  $p_2 = p_0^2, p_1^2, p_2^2, p_3^2, \ldots$

- このとき

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

  となる。ただし N は2つのテキスト(の共通部分)の長さである

パラレル（暗号化）テキストの問題点（2）

- 2つの暗号文 c_1 と c_2 を xor すると、次の関係が得られる

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

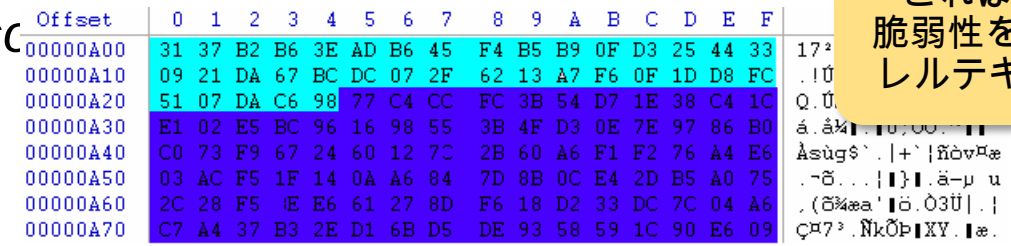
- すると、秘密鍵（つまり疑似乱数系列）とはもはや無関係な次の関係が得られる

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

- この関係式の右辺は2つの平文のビットごとの xor なので、特有な統計情報を持つ

実例

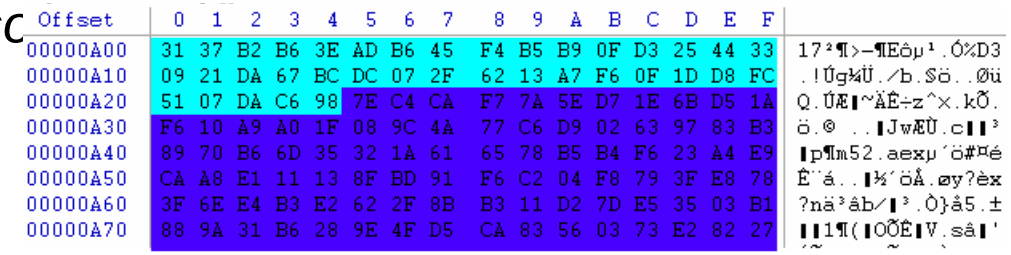
- Word文書を少しだけ変更する（1単語の挿入; 例えば日付の変更）
  - 変更前のテキスト: “*Ceci est un essai de construction de messages parall`eles afin de montrer la vuln ´erabilit ´e du chiffrement de Microsc*



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
00000A10	09	21	DA	67	BC	DC	07	2F	62	13	A7	F6	0F	1D	D8	FC
00000A20	51	07	DA	C6	98	77	C4	CC	FC	3B	54	D7	1E	38	C4	1C
00000A30	E1	02	E5	BC	96	16	98	55	3B	4F	D3	0E	7E	97	86	B0
00000A40	C0	73	F9	67	24	60	12	7C	2B	60	A6	F1	F2	76	A4	E6
00000A50	03	AC	F5	1F	14	0A	A6	84	7D	8B	0C	E4	2D	B5	A0	75
00000A60	2C	28	F5	0E	E6	61	27	8D	F6	18	D2	33	DC	7C	04	A6
00000A70	C7	A4	37	B3	2E	D1	6B	D5	DE	93	58	59	1C	90	E6	09

「これはMicrosoft Wordの暗号化の脆弱性を示すための、(2つの)パラレルテキストの構成テストです」

  - 変更後のテキスト: “*Ceci est un essai de construction de* **deux** *messages parall`eles afin de montrer la vuln ´erabilit ´e du chiffrement de Microsc*



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
00000A10	09	21	DA	67	BC	DC	07	2F	62	13	A7	F6	0F	1D	D8	FC
00000A20	51	07	DA	C6	98	7E	C4	CA	F7	7A	5E	D7	1E	6B	D5	1A
00000A30	F6	10	A9	A0	1F	08	9C	4A	77	C6	D9	02	63	97	83	B3
00000A40	89	70	B6	6D	35	32	1A	61	65	78	B5	B4	F6	23	A4	E9
00000A50	CA	A8	E1	11	13	8F	BD	91	F6	C2	04	F8	79	3F	E8	78
00000A60	3F	6E	E4	B3	E2	62	2F	8B	B3	11	D2	7D	E5	35	03	B1
00000A70	88	9A	31	B6	28	9E	4F	D5	CA	83	56	03	73	E2	82	27

パラレル(暗号化)テキストの検出

- こうしたパラレル性の仮定のもとでは、大量のテキストの中からパラレル暗号化テキストを見つけるのは非常に簡単である:
  - ランダムでないファイルからランダムなファイルを見つけるのと同じくらい簡単
  - 最も基本的な統計テスト
- 暗号化テキストの各ペアの xor を求め、ビット値が 0 になる個数 Z を数える
  - 2つのテキストがパラレルでなければ(例えば異なる鍵で暗号化されていれば)、Zは正規分布 $N(N/2, \sqrt{N}/2)$に従う
  - 2つのテキストがパラレルならば、Zは正規分布 $N(np, \sqrt{p}(1-p))$ に従う。ここで $p>1/2$ であり、p はあるビットが0になる確率である
- このテストは、1時間に数千個のテキストを探索可能
- パラレルテキストの完全な組を検出するために、パラレル性が同値関係であるという事実をしようしているだけである

パラレル（暗号化）テキストの検出

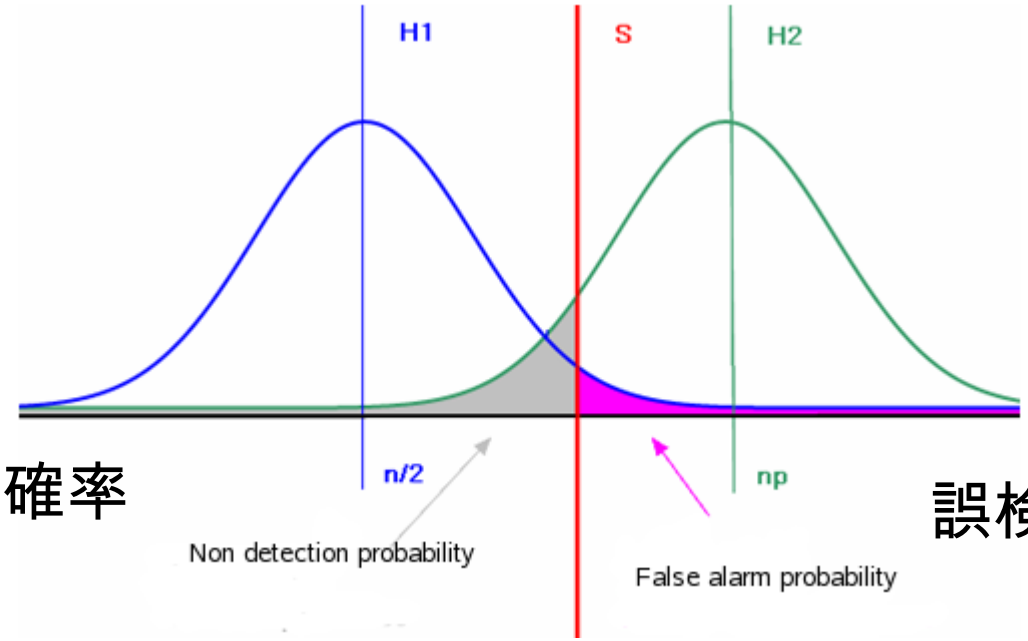
- $Z = \sum_{i=1}^{N}(c_1^i \oplus c_2^i \oplus 1)$ を計算
- Z の極値を見つける

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

- ここではテキスト 1, 2, 3, 4, 5 がパラレルであることがわかる.

パラレル（暗号化）テキストの検出（3）

- 同等な統計テスト
  - 検出しきい値 S を用いて Z の値を選択する



非検出確率

誤検出確率

- このステップは平文が使用している言語に無関係である!!

ターゲット言語の統計モデル

- まずターゲット言語の n-gram コーパス (n文字並びの頻度データベース) を用意する
- 英語はモデル化が最も容易な言語の一つ
- nの最適値は 4 か 5 である (もし4つ以上のパラレルテキストがあれば、n=3 でもうまくいく)

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

- 言語水準や専門性に応じて専用のコーパスを作成することができる
- 初歩的な犯罪科学や諜報学は役立つ

ターゲット言語の統計モデル (2)

- n-gramコーパスの要件
  - 言語水準、文脈、言い回しをとらえている
  - 統計的に適格
  - 文字空間が十分に大きい
- たいていの場合、現代語に基づいて作られた 4-gram コーパスで十分である
- 我々は96文字の文字空間を使用した

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		

- 英語のテキストは、はるかに簡単である

ターゲット言語の統計モデル (3)

- 言語レベルとコーパスへの影響 (質的側面).

```
" 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
```

図: 非現代語文書(左)、現代語文書(中)、現代軍事文書(右)をもとにしたコーパス

- ハッシュテーブルを利用することでメモリ/時間リソースを節約できる

暗号解析の原理

- 最低3つのパラレルテキスト C_1, C_2, C_3 を想定する
- パラレルテキストが2つだけでも攻撃は可能だが、実装がやりにくくなる(意味解析の手続きが必須となる)
- コーパスに含まれる n-gram $T_i = (T^1_i, T^2_i, \ldots, T^n_i)$ の頻度を f_i とし、C_1 の n-gram 平文の候補とする
- この n-gram と、暗号文 C_1 の最初の n-gram (テキストのインデックスを1とする) $(C^1_1, C^2_1, C^3_1, C^4_1)$ の xor を求める。すると以下のような暗号化乱数 n-gram の候補 σ_1が得られる

$$\sigma_1^j = C_1^i \oplus T_i^j \qquad \text{for } i = 1, 2, \ldots n$$

暗号解析の原理(2)

- この暗号化乱数 n-gram の候補σ_1と、暗号文 C_2 と C_3 の最初の暗号文 n-gram とをそれぞれ xor する。するとコーパスにおいて頻度が f_k, f_l の平文 n-gram T_k, T_l と対応する可能性のある2つの平文 n-gram 候補が得られる。
- 3つの頻度を入力とする関数 $Z_i = F(f_i, f_k, f_l)$ を計算する。ここで関数 F() は単調増加関数である。そして最大値 Z_i 値を求める。
- C_1 内の次の暗号文 n-gram に移り、これを C_1,C_2,C_3 の共通部分が終わるまで繰り返す。

一般的なアルゴリズム

Input: m encrypted texts $C_1, \ldots, C_m$. Each C_j is a sequence of n-grams T_j^k. A corpus $T = \{(T_i, f_i)\}$
Output: m plaintexts $P_1, \ldots, P_m$ (sequence of n-grams P_j^k)

```
For every n-gram T_i in T do
    Z ← 0
    For every n-gram T_1^k de C_1 do
        Compute σ_k = T_i ⊕ T_1^k.
        For j from 2 to m do
            Compute M_j^k = σ_k ⊕ C_j^k
            Recover frequencies f_j^k in T
        End For
        If F(f_1^k, ..., f_m^k) > Z Then
            Z = F(f_1^k, ..., f_m^k)
            For j from 1 to m do
                P_j^k = M_j^k
            End For
        End If
    End For
End For
```

基本的な実例

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

図: 正しい平文推定(左)と誤った平文推定(右) ("?"は印字できない文字を示す)

- 明らかに $F(f_1, f_2, f_3) > F(f'_1, f'_2, f'_3)$ なので、左の推定が正しい

重要なパラメータ

- 最終的な成功確率に重大な影響をもたらすパラメータ
  - 頻度関数 *F*
  - 復号方法
  - 決定方法
- 多くの改良により、暗号解析の劇的な高速化と、全テキストを復元する最終的な成功確率の向上が可能となる

頻度関数 F

- 単調増加関数でなければならない
  - 加法型であるか

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

  - 乗法型であるかのいずれか

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

- 乗法型の頻度関数はより効果的である。なぜなら、n-gramの頻度の効果を増幅するからである。一方、まれな周辺度数の(しかし正しい)平文n-gramの効果が抑制されることもある。
- a = 0.3が最適値である

復号方法

- 暗号文から n-gram を取り出す方法に依存する
  - ノーマルモード： n-gram を重なり無く(連続的に)取る方法。この方法はそれほど効率的ではない

  - オーバーラップモード： n-gramは n-1 個の文字を共有する

- オーバラップモードは最適化とアルゴリズム的な技術を適用する余地が非常にあり、もっとも効率的である
- 重なりを持たせることで、最終的な平文 n-gram の信頼性を大いに向上させることができる

復号方法： 基本例

- どうにかして、最もそれらしい組み合わせの復号を行い(量的側面)、一貫性のある復号にする(質的側面)
- (共通する)テキストの最後で復号の成功確率を最適化する

決定方法

- この暗号解析では符号の復号のような操作を行っており、誤り訂正符号の技術が利用できるかもしれない.
- 強気な決定方法： 各 n-gram の候補リストから、ベストなものだけを選択する
  - どのようなものであれ 3-gram エラーは復元が難しく、最終的な平文には数多くの「穴」が生じる
  - 平文に含まれる n-gram（適切な名前, 専門用語,...）が少なすぎると、問題が生じる
- 柔軟な決定方法： 各 n-gram の候補リストから、ベスト *p* (まで)を選択する
  - この場合、前の n-gram の候補リストにおける誤った決定を防ぐことが可能となる(正しい n-gram が2位の場合)
  - 少しトリッキーな実装が必要になるが、はるかに効果的である

微調整と最適化

- 最適なアプローチはこれまでに述べた全ての重要な要素を組み合わせることである.
  - a=0.3 の乗法的頻度関数 F を用いる
  - 全ての最適化を有効にしたオーバーラップモードの使用
  - 柔軟な決定法の使用($5 \leqq p \leqq 10$).
- しかし、他を微調整することで、この暗号解析の効率は向上可能でもある.

微調整と最適化（２）

- 選択した文字空間に含まれない文字を持つ n-gram が現れた場合その、推定は棄却する
- 新しい n-gram を推定したら、逐次的に *m* 個の平文の意味解析する
  - パラレル暗号化テキストが2個しかない場合には必須
  - 対応する際にはいくつかの自由度がある

    THER EISA ROTA TING EFFE CT,

    WHEN DEAL INGW ITHT WOTE XTS

    と

    THER DEAL ROTA TING WOTE XTS

    WHEN EISA INGW ITHT EFFE CT,

    は統計的には同じ解だが、意味的には異なる
  - 意味解析は部分的にのみ有効だが、言語をマルコフ過程と見なすことで、大域的に解析できる。(フランス語は19-重マルコフ過程である)

他の弱点の悪用

- 根本的な問題は、テキストは更新時に前の revision を上書きする（のが普通）という事実である
- だから理想的なOSでは、(パラレル暗号化文書の数の)パラレル数は1にすべきである
- その場合、この暗号解析は実行不能となる
- 求めるべき理想は他にもある:
  - Windows システムには他の（それ自体は罪が無いはずの）弱点がある： それは一時ファイルとその安全でない消去法である
  - パラレル度を増やすのに有効で（時に、非常に重要な方法で）ある
- これら２つを組み合わせることで、強力なフォレンジック解析が可能となる

パラレル度の増加

- 一時ファイル (リビジョンごと

- これら一時ファイルの消去法は安全でない: 復元ソフトが利用できる!!

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

- 平均的には、パラレル度は4～6になる
- これらの更新ファイルは(悪意のある)USBメモリを用いて簡単に行える。これ以上は単なるフォレンジックではなくなる。

実験結果

- (異なる言語グループの)異なる言語における実験を実施した
  - テストグループ1: 共用語 / 非現代語
  - テストグループ2: 共通語 / 現代語
  - テ

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 %

- 全ての最適化を用いれば成功確率を100%に近づけられる
- 適切な名前や珍しい専門用語を扱うにはオペレータによる
  人間チェックが少しだけ必要となる

Excel の場合

- この場合の解読はより難しくなるが、原理は同じである。我々は Word の時と同じくらい効率的に、パラレルテキストからのデータ復元に成功した
  - データ開始位置のオフセットは固定されていない
  - データ構造が全く異なる (テキストでなくセルとなる).
  - データの性質が異なる (文字よりも数字).
  - セルの更新データはシートデータの最後に保存されている
- 以下の事実を利用することで、これらの問題を回避できる
  - データは固定データ 0x8C000400 の31バイト後に開始する
  - 終了マーカーはシート内のセルの個数から求められる。データは固定データ 0xFF001200 + α の直前で終了する。ここで$\alpha = (8 \times p) \times 256$ である。従って、この場合のマーカーは 0xFF000a00, 0xFF001200, 0xFF1a00. . . .となる

Excel における更新

- 次のテキストと、その更新版を考える

	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

- この更新は以下のように保存されている

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	0D	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	0D	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	69	66	69	63	61	74	69	6F	6E	FF	00	modificationÿ

Excel における暗号化の問題

- 次の暗号文と、その更新の暗号文を考える

	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

- 不具合の特定

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	6C	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

Excel のパラレルファイルの検知

- 原理は全く同じである

```
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
```

- Wordの場合と大きな違いはない.

Excel の暗号解析

- やはり原理は同じである
- 追加的に 2 つの制限があるが、対処可能である
  - XX 00 00 という型の特別な(セル)セパレーターフィールドがデータに含まれる

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

  この制限は非常に興味深い。というのも、このデータは平文として登場する可能性が高く、このため、誤った n-gram 予測を生じさせる
  - 特別な n-gram コーパスを使用することになる (文章がない、 文字空間が異なる、動詞はほとんどない、ほとんどは数字...).
- 一般に、Word よりも並列度が高い
- Excel の暗号解析は効率的に操作可能である

成果のまとめ

- (Office 2003 までの) Microsoft Office の暗号を解読する実行可能なテクニック/ツールを開発した.
  - 主にフォレンジックが目的
  - しかしパラレルテキストの盗みだし攻撃を通じて実際に適用可能 (悪意のある USB メモリ, スパイマルウェア...).
- この攻撃は、ストリーム暗号やストリーム暗号的なブロック暗号のモードにおける秘密鍵の誤用（ちょっとしか変更されていないIVでの再利用）を突いている
- そのような誤用事例は予測や予想をはるかに超えている

これはトラップなのか？

- 実に良い質問である！
- ある不具合は他の不具合と組み合わされることで（期待される）トラップになり得る
- 特に２つの不具合が時間経過や (OfficeおよびWindowsの)バージョンを経ても残っている場合
- このようなトラップの作り方に関する考察
  - 何も問題がないように見える２つ以上の不具合を使用せよ
  - 秘密情報分散方式や閾値方式を使用せよ
  - 暗号方式（例えばブロック暗号）を拡張してトラップのある暗号を作り出す
- まだ研究中である

質疑応答

Thanks to Franck Bonnard for his help and his friendship !

- ご静聴ありがとうございます
- 質問は？ ... (馬鹿な質問なんてありませんよ !)...
- 回答します ...(でも馬鹿な回答というのは実際にあるのです).

訳注 - ヤク中 -

スライド33の「マルコフ過程」

- 言語統計処理（コーパス処理）手法らしい
- N-重マルコフ過程で言語を近似する場合
  直前のN-1要素からその次の要素を予測
  （最尤推定）するらしい

> N-gramモデルは，情報理論の創始者として知られるクロード・エルウッド・シャノン（Claude Elwood Shannon 1916-2001）が考案した確率的な言語モデルである．言語の生成過程をN-1重マルコフ過程で近似したものであり，「ある言語単位の系列の中で，言語単位のN個の並びの組み合わせが，どの程度出現するか」を調査する言語モデルである．このモデルによって言語の局所的な特徴を表現することができる

大黒慶久「印刷原稿の多言語識別」, Ricoh Technical Report No.29, 2003
http://www.ricoh.co.jp/technology/techreport/29/pdf/A2904.pdf