ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1999
Seria: INFORMATYKA z. 36 Nr kol.1414

Romuald BŁASZCZYK
Uniwersytet Śląski, Instytut Informatyki

ANALIZA SPRAWNOŚCI METOD KRYPTOGRAFICZNYCH

Streszczenie. W pracy zaprezentowano wyniki badań czasu oraz szybkości szyfrowania programowych realizacji wybranych algorytmów kryptograficznych: FEAL, IDEA, SAFER. Badano tutaj wpływ wielkości pliku, trybu pracy oraz długości klucza na czas i szybkość szyfrowania. Wyniki zebrano w tabelach i przedstawiono na wykresach. Są one podstawą wniosków do dalszych prac w tym zakresie.

ANALYSIS OF THE EFFICIENCY OF CRYPTOGRAPHIC METHODS

Summary. In the paper author presented results of investigation of coding time and coding speed of programmable realisation of selected enciphering algorithms like: FEAL, IDEA, SAFER. Author examined how file size, mode of work and lengths of key affect coding time and coding speed. Results are compiled and presented in tables and on graphs. They constitute ordered base for conclusions to further work in this range.

1. Wstęp

Kryptografia to złożony proces. Wymaga też złożonego procesu kryptoanalizy. Dowodzi tego najlepiej fakt, że wiele metod kryptograficznych po pewnym czasie okazało się metodami mniej bezpiecznymi, niż to wcześniej uważano (np. algorytm IDEA i problem słabych kluczy). Jawi się więc pytanie: czy to taką metodę w ogóle dyskwalifikuje ?

Jak w wielu dziedzinach, tak i w ochronie danych jawi się pytanie: czy wszystko trzeba tak samo chronić ? Czy nie należałoby również i tutaj zastosować pewnej gradacji bezpieczeństwa ? A jeżeli tak, to czym się kierować przy doborze metody zabezpieczenia w odniesieniu do kwestii utajnienia informacji.

Podjęto więc próbę oszacowania sposobu doboru do zabezpieczanych danych adekwatnej metody szyfrowania. W pierwszym etapie dokonano eksperymentalnych badań różnych aspektów implementacyjnych dostępnych metod kryptograficznych. Złożoność tych metod uniemożliwia oszacowanie wpływu tych czynników bez ich doświadczalnego zbadania. W przedsięwzięciu tym dokonano badania takich czynników, jak:

- wpływ wielkości klucza na czas szyfrowania,
- wpływ wielkości klucza na czas kryptoanalizy (analiza teoretyczna),
- wpływ wielkości szyfrowanego zbioru na czas szyfrowania,
- wpływ trybu pracy metody kryptograficznej na szybkość szyfrowania.

W pracy zaprezentowano wyniki badań szybkości szyfrowania programowych realizacji wybranych algorytmów kryptograficznych. Celem tej pracy jest więc prezentacja metod w ujednoliconych warunkach pracy (co bardzo obiektywizuje charakterystyki szybkościowe tych metod) oraz eksperymentalne sprawdzenie wpływu trybów pracy algorytmu na jego sprawność.

Zostanie tutaj zaprezentowana tylko część z przeprowadzonych doświadczeń. Dalsze badania w tym zakresie, jak również propozycja formalnego ujęcia postawionego problemu będą przedmiotem dociekań w następnych etapach.

2. System kryptograficzny

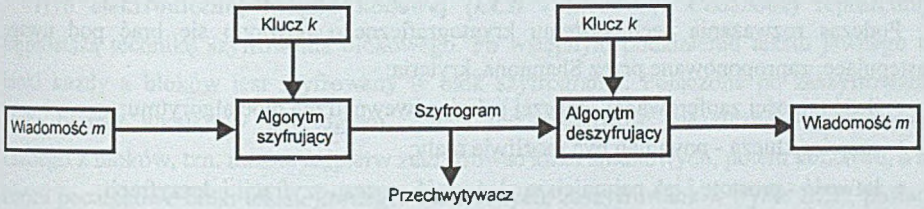
System kryptograficzny [12, 13, 10], zwany też *kryptosystemem*, składa się z następujących elementów:

- przestrzeni wiadomości jawnych M,
- przestrzeni wiadomości zaszyfrowanych C,
- przestrzeni kluczy K,
- przekształcenia szyfrującego e_k określonego algorytmem szyfrowania e oraz kluczem $k \in K$,
- przekształcenia deszyfrującego d_k określonego algorytmem deszyfrującym d oraz kluczem $k \in K$.

Nadawca szyfruje wiadomość jawną m za pomocą e_k. Otrzymana wiadomość zaszyfrowana $c=e_k(m)$ jest przesyłana do odbiorcy i deszyfrowana tam za pomocą d_k. Aby system działał poprawnie, powinna być spełniona zależność:

$$d_k(c)=d_k(e_k(m))=m$$

Ogólny schemat systemu kryptograficznego przedstawia rys. 1.



Rys. 1. System kryptograficzny
Fig.1. Enciphering system

Systemy kryptograficzne można podzielić na dwie klasy: systemy o ograniczonym zasięgu i systemy ogólnego stosowania. Bezpieczeństwo systemu o ograniczonym zasięgu wynika z utrzymania w sekrecie algorytmów szyfrującego i deszyfrującego. We współczesnej kryptografii można implementować systemy kryptograficzne ogólnego stosowania. Ich bezpieczeństwo nie jest uwarunkowane tajnością algorytmów szyfrującego i deszyfrującego, lecz zależy ono od tajnego klucza deszyfrującego.

System kryptograficzny jest *systemem endomorficznym*, jeśli wiadomości jawna i tajna są przedstawione za pomocą tego samego alfabetu.

Uznając dostępność klucza szyfrującego jako kryterium należy wyróżnić dwa typy systemów kryptograficznych: *symetryczny* i *asymetryczny*.

System symetryczny (jednokluczowy, prywatny), np. DES, oparty jest na wspólnym tajnym kluczu, służącym zarówno do szyfrowania, jak i do deszyfrowania.

W *systemie asymetrycznym* (dwukluczowym, publicznym):

- klucz szyfrujący jest inny niż deszyfrujący;
- klucz deszyfrujący nie jest wyznaczany na podstawie klucza szyfrującego;
- każdy kto ma dostęp do klucza publicznego może zaszyfrować wiadomość, odszyfrować może tylko osoba posiadająca odpowiedni klucz deszyfrujący;
- klucz szyfrujący nazywany jest *kluczem publicznym*, a klucz deszyfrujący - *kluczem prywatnym*.

Inne kwestie związane z klasyfikacją metod kryptograficznych nie będą tu poruszane ze względu na tematykę tej pracy.

3. Zasady projektowania szyfrów

Każdy z nowo powstałych systemów kryptograficznych powinien spełniać pewne minimum bezpieczeństwa określone przez Claude Elmwooda Shannona.

Pięć kryteriów Shannona

Podczas rozważania cech systemu kryptograficznego powinno się brać pod uwagę następujące, zaproponowane przez Shannona, kryteria:

- ilość tajności zaoferowanej - inaczej mówiąc, wewnętrzną moc algorytmu;
- rozmiar klucza - powinien być możliwie mały;
- łatwość - prostotę i jak najmniejszą złożoność procesu szyfracji i deszyfracji;
- propagację błędów - powinna być zminimalizowana;
- powiększanie się wiadomości - jest niewskazane.

Oczywiście,podczas projektowania systemu kryptograficznego, który opiera się na języku naturalnym, kryptolog nie jest w stanie „usatysfakcjonować" wszystkich pięciu warunków jednocześnie, gdyż niektóre kryteria wzajemnie się wykluczają.

Projektując system kryptograficzny, należy przyjąć założenia pesymistyczne, tzn. takie, że kryptoanalityk usiłujący złamać szyfr będzie posiadał jak najwięcej informacji możliwych do zdobycia. Szyfr musi być tak zaprojektowany, aby ilość informacji nie miała istotnego ujemnego wpływu na jego bezpieczeństwo. Założenia te nazywane są *warunkami najgorszego przypadku*:

- kryptoanalityk posiada pełną znajomość systemu szyfrującego;
- kryptoanalityk uzyskał dostateczną ilość szyfrogramu;
- kryptoanalityk zna tekst otwarty odpowiadający pewnemu szyfrogramowi.

Proces kryptoanalizy musi być więc na tyle trudny, aby i w takim przypadku jego przeprowadzenie nie należało do zadań prostych.

4. Tryby pracy metod kryptograficznych

Najczęściej stosowane algorytmy blokowe szyfrowania praktycznie implementuje się w jednym z poniższych trybów pracy [12, 13, 10, 14]. Wykonano więc badania pokazujące wpływ poszczególnych trybów na szybkość pracy metod kryptograficznych. Z punktu widzenia celu pracy jest to ważne zagadnienie.

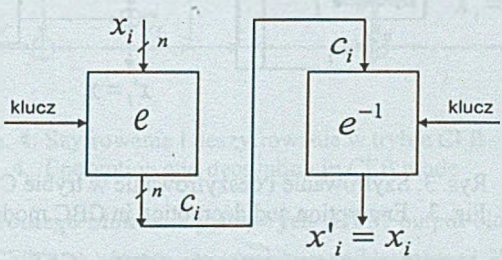
W przedstawionych poniżej trybach szyfrowania wykorzystuje się prostą operację różnicy symetrycznej, która w algorytmie szyfrowania ma postać:

$$c_i = e_{k_i}(m_i) = m_i \oplus k_i$$

Natomiast algorytm deszyfrowania wygląda następująco:

$$m_i = d_{k_i}(c_i) = c_i \oplus k_i = (m_i \oplus k_i) \oplus k_i = m_i$$

Tryb **elektronicznej książki kodowej** (*ECB - Electronic Codebook*) reprezentuje najprostszą technikę szyfrowania blokowego. Po wstępnym podzieleniu tekstu jawnego na bloki każdy z bloków jest szyfrowany w blok szyfrogramu. Połączone po zaszyfrowaniu bloki tworzą wynikowy kryptogram. Zaletą jest możliwość niezależnego szyfrowania każdego z bloków, tzn. można najpierw zaszyfrować kilka środkowych, potem końcowe, a na końcu początkowe bloki tekstu jawnego. Baza danych, zaszyfrowana w trybie ECB, posiada możliwość deszyfracji każdego rekordu niezależnie od pozostałych. Technika ta nie jest jednak dostatecznie bezpieczna. Jeżeli kryptoanalityk zna tekst jawny odpowiadający danemu blokowi szyfrogramu (najczęściej ośmioznakowy - 64 bity), wówczas łatwo może przeprowadzić analizę wyszukując sekwencje te w tekście zaszyfrowanym.



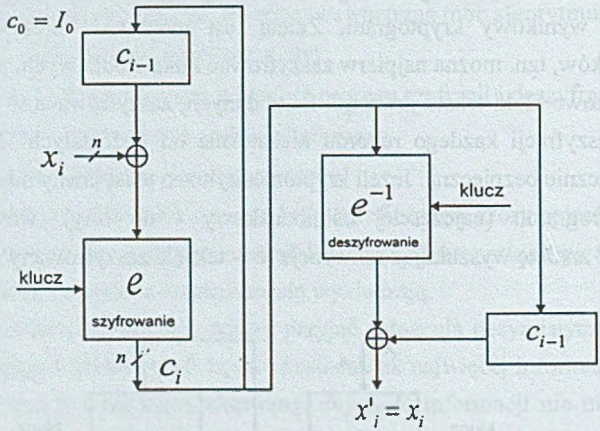
Rys. 2. Szyfrowanie i deszyfrowanie w trybie ECB
Fig. 2. Encryption and decryption in ECB mode

Tryb **wiązania bloków zaszyfrowanych** (*CBC - cipher block chaining*) jest odporny na rodzaj ataku przedstawiony powyżej. Każdy blok tekstu jawnego jest przed szyfrowaniem łączony operacją XOR z poprzednim blokiem szyfrogramu. Wykorzystano w tym trybie mechanizm sprzężenia zwrotnego, tzn. blok wyjściowy jest wykorzystywany do modyfikacji kolejnego bloku wejściowego. Początkowa wartość rejestru sprzężenia zwrotnego I_0 (*wektor początkowy*) jest inicjowana przez użytkownika i powinna być losowa. Każdy blok szyfrogramu jest zależny zarówno od bieżących bloków tekstu jawnego, jak i od wszystkich poprzednich bloków. Podczas deszyfracji blok szyfrogramu jest odszyfrowywany i zapamiętywany w rejestrze sprzężenia zwrotnego, który swoją zawartość będzie sumował modulo 2 z kolejnym odszyfrowanym blokiem. Operacje szyfrowania i deszyfracji wyglądają następująco:

$$c_i = e_k(m_i \oplus c_{i-1})\ ;\ C_0 = I_0$$

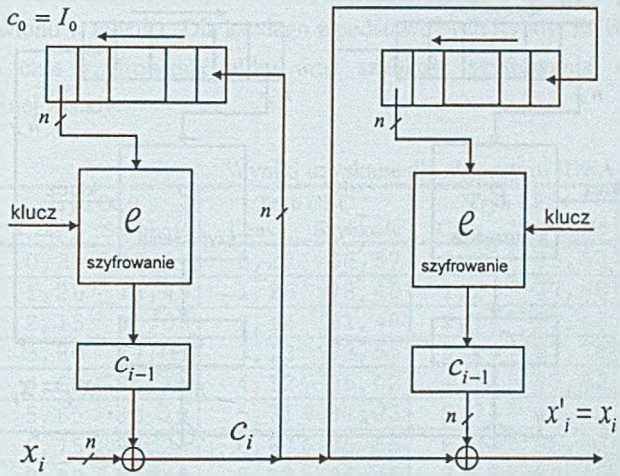
$$m_i = d_k(c_i) \oplus c_{i-1} = d_k(e_k(m_i \oplus c_{i-1})) \oplus c_{i-1} = (m_i \oplus c_{i-1}) \oplus c_{i-1} = m_i$$

Wadą CBC jest możliwość wpływania błędu, który wystąpi podczas szyfrowania bloku na wszystkie kolejne bloki. W zastosowaniach sieciowych nie można rozpocząć deszyfrowania, dopóki ostatni blok szyfrogramu nie zostanie przysłany.



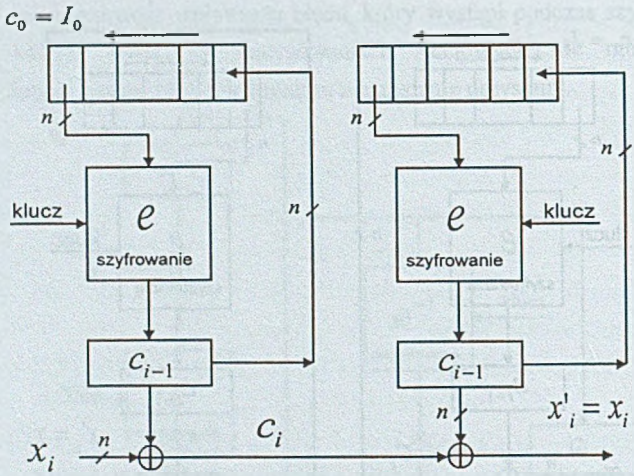
Rys. 3. Szyfrowanie i deszyfrowanie w trybie CBC
Fig. 3. Encryption and decryption in CBC mode

Tryb **sprzężenia zwrotnego zaszyfrowanego tekstu** (***CFB*** - *cipher feedback mode*) umożliwia szyfrowanie danych w jednostkach mniejszych niż rozmiar pobieranego bloku tekstu jawnego. Pojawia się możliwość szyfrowania danych w porcjach jednobitowych, stosując tryb 1-bitowego CFB. Najpierw rejestr przesuwający sprzężenia zwrotnego (LFSR) inicjowany jest wartością początkową I_0, podobnie jak w trybie CBC. Dla trybu n - bitowego, n najstarszych bitów rejestru przesuwającego zostaje zaszyfrowanych kluczem k. Wynik szyfrowania c_{i-1} jest sumowany modulo 2 z n bitami tekstu jawnego. Rezultat tej operacji c_i jest umieszczany na n najmłodszych bitach rejestru przesuwającego, które rugują pobrane wcześniej bity. Jednocześnie blok ten jest blokiem wynikowego szyfrogramu. Dla kolejnej iteracji pobiera się nowych n bitów, które po zaszyfrowaniu będą sumowane modulo 2 z kolejnym blokiem tekstu jawnego. Po obu stronach, szyfrowania i odszyfrowania, algorytm blokowy stosuje się w trybie szyfrującym. Tryb CFB jest przykładem samosynchronizującego się szyfru strumieniowego.



Rys. 4. Szyfrowanie i deszyfrowanie w trybie CFB
Fig. 4. Encryption and decryption in CFB mode

Tryb **sprzężenia zwrotnego bloków wyjściowych** (**OFM** output-block feedback mode) jest podobny do trybu CFB z tą różnicą, że zamiast wyniku operacji XOR na najmłodsze pozycje rejestru przesuwającego kierowany jest c_{i-1}, tzn. wynik szyfrowania kluczem k bloku najstarszych bitów rejestru. Tryb ten nosi czasem nazwę trybu **wewnętrznego sprzężenia zwrotnego**, ponieważ sprzężenie zwrotne nie zależy od bloków tekstu jawnego, lecz jedynie od wartości rejestru przesuwnego. Tryb ten jest przykładem synchronicznego szyfru strumieniowego. Wektory początkowe I_0 powinny być różne dla każdej wiadomości podczas wykorzystania tego samego klucza.



Rys. 5. Szyfrowanie i deszyfrowanie w trybie OFM
Fig. 5. Encryption and decryption in OFM mode

5. Wyniki prowadzonych badań

Badania przeprowadzono dla metod IDEA [1, 8], FEAL [5, 7, 9], SAFER [6, 10] oraz dodatkowo dla metody ElGamala [3, 4].

Testy poszczególnych algorytmów szyfrowania przeprowadzono na komputerze klasy PC Pentium 90 MHz, chociaż nie ma to większego znaczenia, gdyż wyniki te są porównywane między sobą. Problemy tu przedstawione ograniczono do procesu szyfrowania, gdyż otrzymane czasy szyfrowania i deszyfrowania były podobne. Kryteria, jakimi się posługiwano przy ocenie algorytmów, to **czas szyfrowania** oraz **szybkość szyfrowania**.

Czas szyfrowania [s] jest określany z wykorzystaniem RAM – dysku. Umożliwia to pominięcie czasu dostępu do dysku twardego. Otrzymane dane zostały przedstawione z dokładnością do setnych części sekundy.

Szybkość szyfrowania [kB/s] jest uzyskiwana poprzez podzielenie wielkości pliku przez czas jego szyfrowania. Należy brać tu pod uwagę, że im jest ona większa, tym algorytm jest lepszy. Oznacza to, że potrafi zaszyfrować więcej danych w krótszym czasie.

W dalszej części podrozdziału przedstawiono wyniki otrzymanych testów w postaci tabel i wykresów. Otrzymane wyniki zestawiono na dwa sposoby:

- czas szyfrowania danego algorytmu w czterech trybach pracy,
- czas szyfrowania algorytmów w danym trybie.

Po przeprowadzeniu testów szybkości szyfrowania algorytmem IDEA otrzymano wyniki, które przedstawiono w tabeli 1. Dla każdego z podstawowych trybów ECB, CBC, CFB, OFB przedstawiono czas szyfrowania pliku oraz szybkość szyfrowania, obliczoną według wcześniej podanej zasady.

Tabela 1

Wyniki uzyskane dla algorytmu IDEA

Rozmiar plików [kB]	Tryb ECB		Tryb CBC		Tryb CFB		Tryb OFM	
	Czas	Szybkość	Czas	Szybkość	Czas	Szybkość	Czas	Szybkość
9,88	0,28	35,27	0,27	36,57	0,27	36,57	0,22	44,89
98,90	1,26	78,49	1,26	78,49	1,27	77,88	1,21	81,74
195,76	2,15	91,05	2,14	91,48	2,14	91,48	2,14	91,48
292,80	3,80	77,05	3,13	93,55	3,14	93,25	3,80	77,05
391,48	4,60	85,11	4,12	95,02	4,12	95,02	4,70	83,29
498,00	5,65	88,14	5,71	87,22	5,71	87,22	5,66	87,99
656,40	6,75	97,24	6,87	95,55	6,87	95,55	6,76	97,10
788,94	8,80	89,65	8,18	96,45	8,18	96,45	8,70	90,68
879,43	9,10	96,64	9,12	96,43	9,12	96,43	9,10	96,64
1015,00	10,44	97,22	10,54	96,30	10,60	95,75	10,44	97,22
1494,50	16,59	90,08	16,81	88,91	16,81	88,91	16,64	89,81
1975,10	19,99	98,80	20,27	97,44	20,22	97,68	19,99	98,80
2429,10	26,92	90,23	27,19	89,34	27,19	89,34	26,91	90,27
2939,77	32,57	90,26	32,90	89,35	32,90	89,35	32,52	90,40
3435,92	34,72	98,96	35,10	97,89	35,90	95,71	34,72	98,96
3925,71	43,45	90,35	43,89	89,44	43,94	89,34	43,45	90,35

Analizując powyższą tabelę, można zauważyć, że otrzymane wyniki szybkości nie dają jednoznacznych wartości, lecz wykazują pewną oscylację, co jest spowodowane niedoskonałością sprzętu, na którym przeprowadzano testy. Szybkość szyfrowania testowanych plików, niezależnie od rodzaju i trybu szyfrowania, osiągała wartość rzędu 85-95 kB/s.

W przypadku małych plików (<200kB) szybkość szyfrowania znacznie spada, a większych (od 200 kB) utrzymuje się na tym samym poziomie. Jest to uwarunkowane czynnościami związanymi z wyznaczaniem podkluczy w części początkowej algorytmu.

Porównując szybkość szyfrowania dla każdego trybu, zgodnie z oczekiwaniami, ECB (Tryb Elektronicznej Książki Kodowej) jest najszybszy, albowiem nie wykonuje się w nim żadnych dodatkowych czynności. Pozostałe tryby osiągają tylko minimalnie gorszą szybkość szyfrowania rzędu 0,1-1,5 kB/s, gdyż operacja XOR (będąca integralną częścią tych trybów) ginie w porównaniu z ilością operacji stosowanych podczas szyfrowania. Co ciekawe, tryb OFM zdaje się nawet uzyskiwać lepszy średni czas szyfrowania od CBC i CFB oraz jest zbliżony do ECB, jako że powoduje on podwyższenie bezpieczeństwa w porównaniu z ECB.

W oparciu o przeprowadzone testy szybkości można zaproponować tryb OFM jako najlepszy dla algorytmu IDEA.

W tabeli 2 przedstawiono wyniki testów szybkości szyfrowania algorytmem FEAL-8 w czterech trybach pracy tej samej grupy plików.

Tabela 2

Wyniki uzyskane dla algorytmu FEAL

Rozmiar plików[kB]	Tryb ECB		Tryb CBC		Tryb CFB		Tryb OFM	
	Czas	Szybkość	Czas	Szybkość	Czas	Szybkość	Czas	Szybkość
9,88	0,22	44,89	0,22	44,89	0,22	44,89	0,22	44,89
98,90	1,40	70,64	1,50	65,93	1,40	70,64	1,40	70,64
195,76	1,87	104,68	1,92	101,96	1,92	101,96	1,87	104,68
292,80	2,86	102,38	2,91	100,62	2,91	100,62	2,85	102,74
391,48	3,79	103,29	3,79	103,29	3,85	101,68	3,79	103,29
498,00	4,62	107,79	4,67	106,64	4,67	106,64	4,61	108,03
656,40	6,50	100,98	6,10	107,61	6,90	95,13	6,40	102,56
788,94	7,20	109,57	7,31	107,93	7,30	108,07	7,25	108,82
879,43	8,34	105,45	8,40	104,69	8,41	104,57	8,35	105,32
1015,00	9,18	110,57	9,22	110,09	9,23	109,97	9,17	110,69
1494,50	13,5	111,03	13,51	110,62	13,6	110,13	13,4	111,53
1975,10	18,6	106,36	18,68	105,73	18,7	105,45	18,6	106,42
2429,10	22,7	107,10	22,85	106,31	22,9	106,31	22,7	107,06
2939,77	27,4	107,25	27,57	106,63	27,6	106,40	27,4	107,49
3435,92	30,9	111,12	31,20	110,13	31,3	109,95	31,0	110,91
3925,71	36,5	107,49	36,75	106,82	36,8	106,68	36,5	107,47

W przypadku algorytmu FEAL-8 uzyskano odmienne wyniki niż we wcześniejszym teście szybkości. Te same pliki są już szyfrowane ze średnią prędkością rzędu 100-110 kB/s. Algorytm FEAL-8 jest więc szybszym algorytmem niż IDEA.

Pozostałe spostrzeżenia i uwagi pokrywają się z tymi zauważonymi przy testowaniu algorytmem IDEA. Otóż małe pliki są również wolniej szyfrowane, a tryb ECB jest również trybem, którym szyfrowanie odbywa się najszybciej. Pozostałe tryby szyfrują minimalnie wolniej, rzędu 0,5-5 kB/s. Tryb OFM jest również niewiele gorszy od ECB, a lepszy od CBC i CFB. Zatem w algorytmie FEAL-8 podobnie jak w IDEA, tryb OFM (Tryb Sprzężenia Zwrotnego Bloków Wyjściowych) może być z powodzeniem stosowany, ponieważ zapewnia większe bezpieczeństwo niż ECB, przy prawie takim samym czasie szyfrowania.

W tabeli 3 przedstawiono testy szybkości ostatniego z algorytmów omawianych w tej pracy SAFER SK-64.

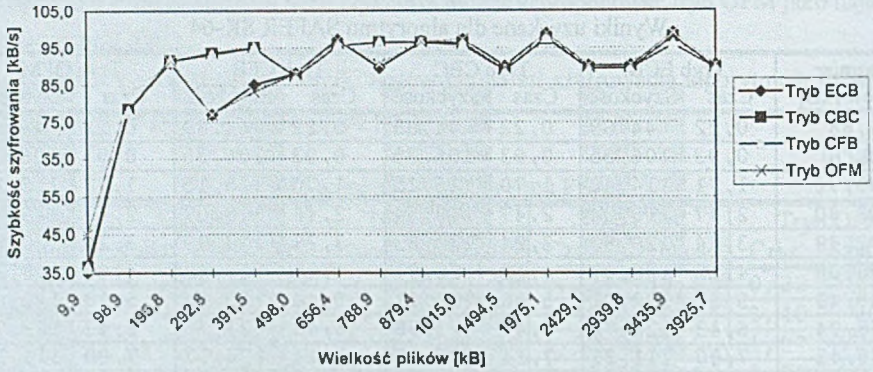
Tabela 3

Wyniki uzyskane dla algorytmu SAFER SK-64

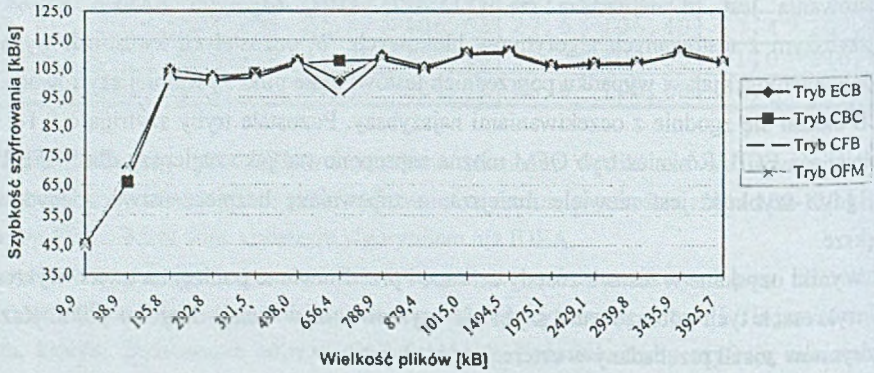
Rozmiar plików [kB]	Tryb ECB		Tryb CBC		Tryb CFB		Tryb OFM	
	Czas	Szybkość	Czas	Szybkość	Czas	Szybkość	Czas	Szybkość
9,88	0,22	44,89	0,22	44,89	0,22	44,89	0,22	44,89
98,90	0,93	106,35	0,93	106,35	0,93	106,35	0,93	106,35
195,76	1,71	114,48	1,70	115,15	1,70	115,15	1,70	115,15
292,80	2,47	118,54	2,47	118,54	2,48	118,07	2,47	118,54
391,48	3,24	120,83	3,24	120,83	3,29	118,99	3,24	120,83
498,00	4,12	120,87	4,12	120,87	4,17	119,42	4,60	108,26
656,40	5,38	122,01	5,38	122,01	5,44	120,66	5,38	122,01
788,94	6,43	122,70	6,48	121,75	6,48	121,75	6,43	122,70
879,43	7,90	111,32	7,14	123,17	7,14	123,17	7,90	111,32
1015,00	8,13	124,85	8,24	123,18	8,24	123,18	8,13	124,85
1494,50	11,92	125,38	12,30	121,50	12,30	121,50	11,92	125,38
1975,10	15,71	125,72	15,87	124,46	15,88	124,38	15,71	125,72
2429,10	19,22	126,38	19,39	125,28	19,38	125,34	19,22	126,38
2939,77	23,24	126,50	23,40	125,63	23,40	125,63	23,24	126,50
3435,92	27,41	125,35	27,63	124,35	27,68	124,13	27,41	125,35
3925,71	30,98	126,72	31,20	125,82	31,19	125,86	30,98	126,72

Porównując otrzymane wyniki z poprzednimi, można zauważyć, że średnia szybkość szyfrowania jest tu najwyższa rzędu 115-125 kB/s. Algorytm SAFER SK-64 jest najszybszym z testowanych algorytmów blokowych. W pozostałych kwestiach wysunięto podobne wnioski jak w wypadku poprzednich testów; małe pliki są wolniej szyfrowane, tryb ECB okazał się zgodnie z oczekiwaniami najszybszy. Pozostałe tryby szyfrują o 0,1-2 kB/s wolniej niż ECB. Również tryb OFM można zaproponować jako najlepszy dla SAFER SK-64, gdyż szybkość jest niewiele mniejsza, a zapewniane bezpieczeństwo zdecydowanie większe.

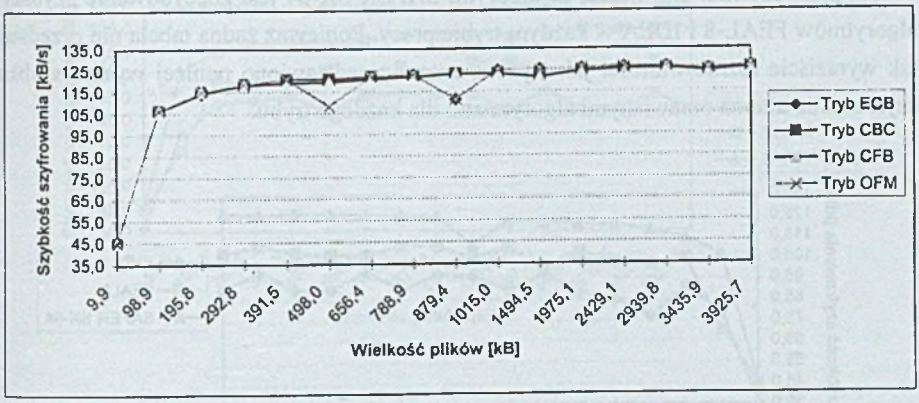
Wyniki uzyskane w testach zostały zebrane i przedstawione poniżej na trzech wykresach. Na wykresach tych zobrazowano **szybkość szyfrowania** w funcji długości pliku. Każdy z algorytmów został przebadany w czterech trybach pracy.



Rys. 6. Porównanie szybkości szyfrowania algorytmem IDEA w czterech trybach pracy
Fig. 6. Comparison of enciphering speeds for IDEA algorithm in four modes of work



Rys. 7. Porównanie szybkości szyfrowania algorytmem FEAL w czterech trybach pracy
Fig. 7. Comparison of enciphering speeds for FEAL algorithm in four modes of work



Rys.8. Porównanie szybkości szyfrowania algorytmem SAFER SK-64 w czterech trybach pracy

Fig.8. Comparison of enciphering speeds for SAFER SK-64 algorithm for four modes of work

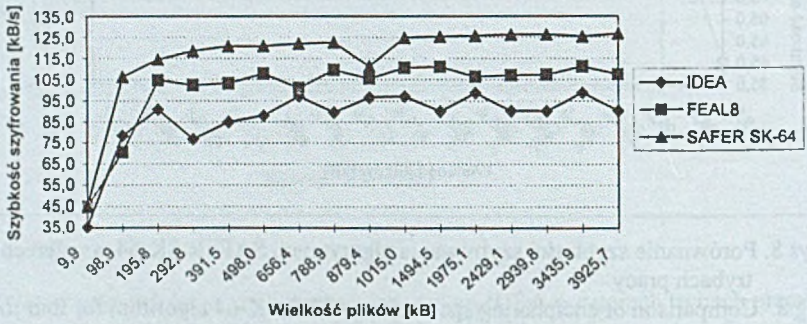
Aby w pełni zobrazować szybkość szyfrowania algorytmami IDEA, FEAL-8 i SAFER SK-64, należałoby je zestawić obok siebie. Tabela 4 przedstawia takie zestawienie, które ukazuje różnice szybkości szyfrowania każdego algorytmu w każdym z trybów.

Tabela 4

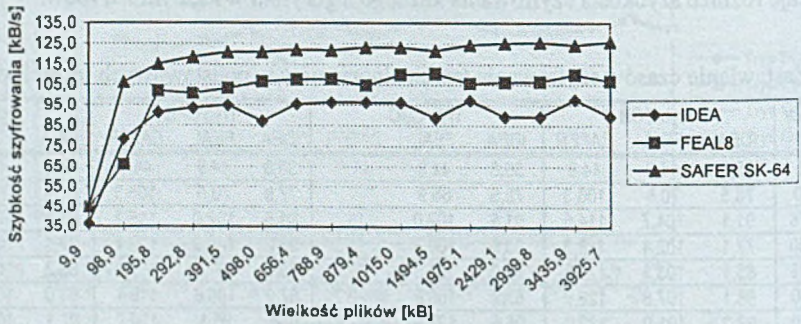
Zestawienie czasów szyfrowania trzech algorytmów w podstawowych trybach pracy

Rozmiar	Tryb ECB			Tryb CBC			Tryb CFB			Tryb OFM		
pliku	IDEA	FEAL	SAFER	IDEA	FEAL	SAFER	IDEA	FEAL	SAFER	IDEA	FEAL	SAFER
9,88	35,3	44,9	44,9	36,6	44,9	44,9	36,6	44,9	44,9	44,9	44,9	44,9
98,90	78,5	70,6	106,3	78,5	65,9	106,3	77,9	70,6	106,3	81,7	70,6	106,3
195,76	91,1	104,7	114,5	91,5	102,0	115,2	91,5	102,0	115,2	91,5	104,7	115,2
292,80	77,1	102,4	118,5	93,5	100,6	118,5	93,2	100,6	118,1	77,1	102,7	118,5
391,48	85,1	103,3	120,8	95,0	103,3	120,8	95,0	101,7	119,0	83,3	103,3	120,8
498,00	88,1	107,8	120,9	87,2	106,6	120,9	87,2	106,6	119,4	88,0	108,0	108,3
656,40	97,2	101,0	122,0	95,5	107,6	122,0	95,5	95,1	120,7	97,1	102,6	122,0
788,94	89,7	109,6	122,7	96,4	107,9	121,7	96,4	108,1	121,7	90,7	108,8	122,7
879,43	96,6	105,4	111,3	96,4	104,7	123,2	96,4	104,6	123,2	96,6	105,3	111,3
1015,00	97,2	110,6	124,8	96,3	110,1	123,2	95,8	110,0	123,2	97,2	110,7	124,8
1494,50	90,1	111,0	125,4	88,9	110,6	121,5	88,9	110,1	121,5	89,8	111,5	125,4
1975,10	98,8	106,4	125,7	97,4	105,7	124,5	97,7	105,5	124,4	98,8	106,4	125,7
2429,10	90,2	107,1	126,4	89,3	106,3	125,3	89,3	106,3	125,3	90,3	107,1	126,4
2939,77	90,3	107,3	126,5	89,4	106,6	125,6	89,4	106,4	125,6	90,4	107,5	126,5
3435,92	99,0	111,1	125,4	97,9	110,1	124,4	95,7	109,9	124,1	99,0	110,9	125,4
3925,71	90,4	107,5	126,7	89,4	106,8	125,8	89,3	106,7	125,9	90,4	107,5	126,7

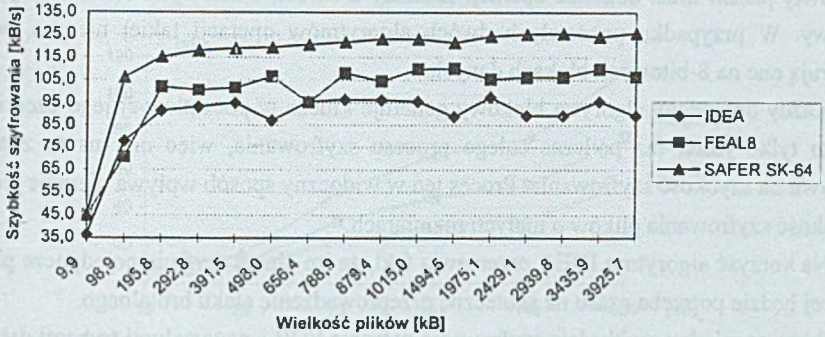
Na pierwszy rzut oka widać, że algorytm SAFER SK-64 jest zdecydowanie szybszy od algorytmów FEAL-8 i IDEA w każdym trybie pracy. Ponieważ żadna tabela nie przedstawia tak wyraziście różnic wartości jak wykres, zatem przedstawiono poniżej wykres szybkości szyfrowania trzema omawianymi algorytmami dla każdego trybu.



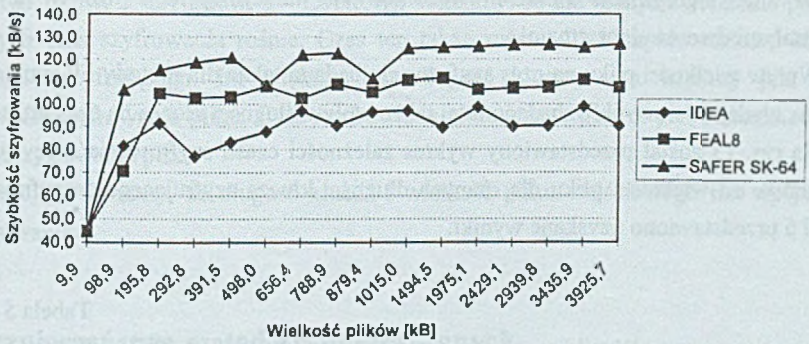
Rys. 9. Szybkość szyfrowania dla trybu ECB
Fig. 9. Encryption speed for ECB mode



Rys. 10. Szybkość szyfrowania dla trybu CBC
Fig. 10. Encryption speed for CBC mode



Rys. 11. Szybkość szyfrowania dla trybu CFB
Fig. 11. Encryption speed for CFB mode



Rys. 12. Szybkość szyfrowania dla trybu OFM
Fig. 12. Encryption speed for OFM mode

W każdym z podstawowych trybów szybkość szyfrowania algorytmem SAFER SK-64 jest większa od prędkości szyfrowania algorytmem FEAL-8 średnio o około 15 kB/s, a od algorytmu IDEA o około 30 kB/s. Algorytm FEAL-8 jest szybszy o około 15 kB/s od algorytmu IDEA. Wynika to z kilku powodów. Dzieje się tak za sprawą wewnętrznej budowy algorytmu. Algorytm IDEA używa 8 pełnych cykli i jednego niepełnego, a FEAL-8 używa 8 cykli. Natomiast pozornie najszybszy algorytm SAFER SK-64 został zaimplementowany tylko z 6 cyklami. Kolejnym elementem spowalniającym algorytm IDEA jest fakt, że używa on 16-bitowych bloków danych, więc za każdym pobraniem 8-

bajtowej paczki musi dokonać operacji zamiany z dwóch 8-bitowych bloków na jeden 16-bitowy. W przypadku pozostałych dwóch algorytmów operacji takiej nie ma, ponieważ szyfrują one na 8-bitowych blokach danych.

Każdy omawiany algorytm blokowy generuje klucze w początkowej jego części. Dzieje się to tylko jeden raz podczas całego procesu szyfrowania, więc nie ma to znaczącego wpływu na szybkość szyfrowania. Proces ten w widoczny sposób wpływa ujemnie jedynie na szybkość szyfrowania plików o małych rozmiarach.

Na korzyść algorytmu IDEA przemawia fakt, że im dłużej szyfruje pojedyncze pliki, tym więcej będzie potrzeba czasu na skuteczne przeprowadzenie ataku brutalnego.

Różnice między szybkością szyfrowania w trybie ECB a pozostałymi trybami dają ogólne informacje o ilości operacji przeprowadzanych podczas szyfrowania. Otóż w przypadku algorytmu IDEA wynoszą 0,1-1,5 kB/s, dla FEAL-8 0,5-5 kB/s, a dla SAFER SK-64 wynosi 0,1-2kB/s. Algorytm FEAL-8 posiada największe różnice, stąd można wnioskować, że kluczowa operacja XOR w pozostałych trybach wpływa bardziej na wynikową szybkość szyfrowania niż w algorytmie IDEA czy SAFER SK-64. Inaczej mówiąc, podczas szyfrowania algorytmem FEAL-8 wykonuje się mniej operacji niż w szyfrowaniu pozostałymi dwoma algorytmami.

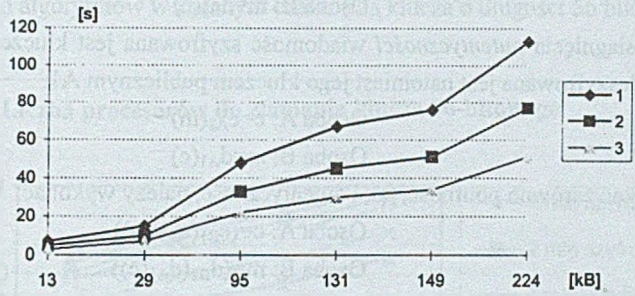
Wpływ wielkości pliku na czas szyfrowania badano także dla metody ElGamala. Oprócz badań „wydajnościowych", zbadano tutaj także wpływ długości klucza na czas szyfrowania.

Na rys. 13 został przedstawiony wykres zależności czasu szyfrowania/deszyfrowania w zależności od wielkości pliku dla różnych długości klucza szyfrującego/deszyfrującego. W tabeli 5 przedstawiono uzyskane wyniki.

Tabela 5

Wyniki przeprowadzonych badań dla metody ElGamala

Nr	Wielkość pliku [kB]	Długość klucza 13 bitów		Długość klucza 22 bity		Długość klucza 31 bitów	
		Szyfr.	Deszyfr.	Szyfr.	Deszyfr.	Szyfr.	Deszyfr.
1	13	3,2 s	3,2 s	5,1 s	5,1 s	7,2 s	7,2 s
2	29	7,0 s	6,9 s	10,4 s	10,3 s	15,2 s	15,2 s
3	95	22,5 s	22,4 s	33,6 s	33, 5 s	48,5 s	48,4 s
4	131	30,8 s	30,7 s	45,1 s	45,0 s	67,8 s	66, 8s
5	149	34,3 s	34,3 s	51,3 s	51,3 s	76,2 s	76,1 s
6	224	51,8 s	51,7 s	77,5 s	77,4 s	113,3 s	113,1 s



Rys.13. Wykres zależności czasu szyfrowania/deszyfrowania od wielkości pliku dla różnych wartości klucza szyfrującego/deszyfrującego

Fig. 13. Graphs of dependences encryption/decryption time on size of file for different length key

Wykres przedstawia trzy różne wartości długości klucza szyfrującego / deszyfrującego (są to: 1 - 31 bitów, 2 - 27 bitów, 3 - 13 bitów). Widzimy, że dla różnych wielkości plików wejściowych czas szyfrowania rośnie. Czas ten także rośnie dla różnych wartości klucza szyfrującego/deszyfrującego. Im większa jest długość klucza, tym większy jest czas szyfrowania/deszyfrowania pliku. Czas szyfrowania pliku jest w dużym przybliżeniu równy czasowi deszyfrowania pliku.

Reasumując, widzimy, że długość klucza i wielkość pliku wejściowego ma wpływ na czas szyfrowania.

6. Bezpieczeństwo metod kryptograficznych

Celem istnienia systemów kryptograficznych jest zapewnienie *poufności* i *autentyczności* przesyłanych bądź przechowywanych danych. Dodatkowym zadaniem może być jeszcze zapewnienie *niezaprzeczalności*.

Jeśli obie strony wykorzystujące systemy jednokluczowe (*symetryczne*) są jednakowo godne zaufania, wówczas we wzajemnej komunikacji mogą sobie zapewnić zarówno poufność, jak i autentyczność. Systemy te nie zapewniają jednak niezaprzeczalności.

W *systemie asymetrycznym* (dwukluczowym, publicznym) należałoby dokonać oddzielnych przekształceń dla zachowania poufności i oddzielnych dla zachowania autentyczności. W celu uzyskania *poufności* nadawca A szyfruje wiadomość kluczem publicznym B1 odbiorcy B, natomiast odbiorca B deszyfruje ją przy użyciu własnego klucza prywatnego B2. Przekształcenia te można zapisać następująco:

Osoba A: $c=e_{B1}(m)$

Osoba B: $m=d_{B2}(c)$

W celu osiągnięcia *autentyczności* wiadomość szyfrowana jest kluczem prywatnym A2 nadawcy A, deszyfrowana jest natomiast jego kluczem publicznym A1.

Osoba A: $c=e_{A2}(m)$

Osoba B: $m=d_{A1}(c)$

Aby uzyskać zarówno poufność, jak i autentyczność, należy wykonać:

Osoba A: $c=(e_{B1}(e_{A2}(m))$

Osoba B: $m=(d_{B2}(d_{A1}(c))$

System kryptograficzny powinien być taki, aby bezpieczeństwo systemu zależało tylko od zagwarantowania poufności kluczy, nie zaś od poufności algorytmów szyfrującego i deszyfrującego: *e* i *d*. Oznacza to, że algorytmy te powinny być na tyle wewnętrznie silne, aby znajomość metody szyfrowania nie determinowała możliwości złamania klucza.

Za algorytm bezpieczny uważa się algorytm, którego złamanie nie jest możliwe. Natomiast algorytm obliczeniowo bezpieczny to taki, który nie może być złamany za pomocą dostępnych środków. Teoretycznie wszystkie algorytmy są możliwe do złamania. Przez system doskonały rozumiemy taki system kryptograficzny, dla którego nie ma lepszej metody złamania, jak próbowanie każdego możliwego klucza. Jest to tzw. łamanie brutalne. Bezpieczeństwo systemu spoczywa więc w kluczu. Im jest on większy, tym trudniej odnaleźć jego wartość, także metodą łamania brutalnego.

Jeżeli np. długość klucza ma 8 bitów, to występuje $2^8 = 256$ możliwych kluczy. Zatem potrzeba 256 prób, aby znaleźć właściwy klucz. Jeżeli znamy szybkość komputera (w operacjach na sekundę), to bardzo prosto możemy wyliczyć maksymalny czas łamania brutalnego na znalezienie właściwego klucza:

$T_{LB}(s^p)$ = {liczba możliwych postaci klucza} * {czas potrzebny na sprawdzenie poprawności klucza}

Uwzględniając zaś, że proces ten wykonywany będzie na m procesorach jednocześnie:

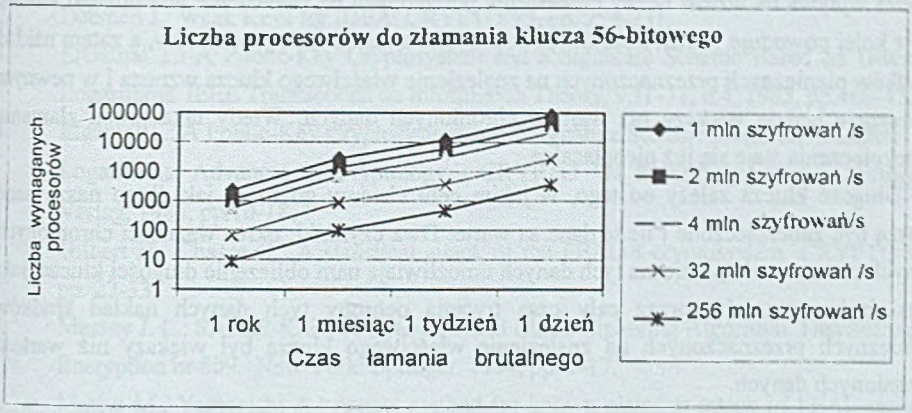
$$T_{LB}(s^m) = T_{LB}(s^p)/m$$

Liczba możliwych postaci klucza może być wyliczona jako: $K_v = 2^l$, gdzie l jest długością klucza.

Czas potrzebny na sprawdzenie poprawności klucza może być określony jako czas szyfrowania według zadanego algorytmu z wygenerowanym kluczem; oznaczmy ten czas przez t_e. Tak więc czas łamania brutalnego na *m* procesorach wynosił będzie

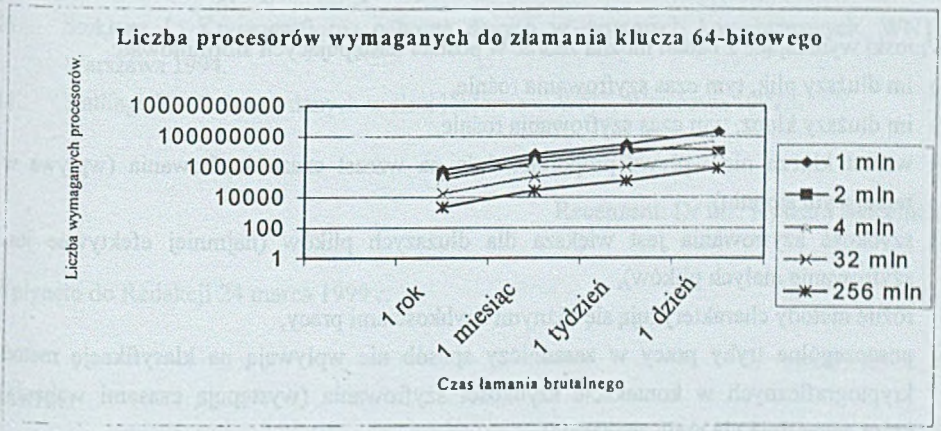
$$T_{LB}(s^m) = \frac{2^l * t_e}{m}.$$

Wykresy na rys.14 i 15 pokazują liczbę procesorów wymaganych do realizacji łamania brutalnego różnych algorytmów w zadanym czasie (dla klucza o długości 56 bitów i 64 bitów odpowiednio).



Rys.14. Liczba procesów wymaganych do realizacji łamania brutalnego różnych algorytmów w zadanym czasie (przy szybkościach pojedynczych procesorów zaznaczonych kolorami)

Fig. 14. Number of processors to making brutal attack of different algorithms in given time (the speed of single processor are marked difrent colors)



Rys. 15. Liczba procesów wymaganych do (skutecznej) realizacji łamania brutalnego różnych algorytmów w zadanym czasie (przy szybkościach pojedynczych procesorów zaznaczonych kolorami)

Fig. 15. Number of processors to making brutal attack of different algorithms in given time (the speed of single processor are marked difrent colors)

Powyższe wykresy najlepiej obrazują właściwy dobór długości klucza w kontekście okresu ważności klucza, który ma zapewnić bezpieczeństwo danych. Zwiększanie długości klucza wpływa na wzrost liczby procesorów potrzebnych na znalezienie właściwego klucza. To z kolei powoduje wzrost kosztów potrzebnych na zakup tych procesorów, a zatem nakład środków pieniężnych przeznaczonych na znalezienie właściwego klucza wzrasta i w pewnym momencie będzie większy niż wartość chronionych danych. Wtedy taka próba złamania zabezpieczenia staje się już nieopłacalna.

Długość klucza zależy od tego, w jakim celu chcemy go użyć, jak długo nasze dane muszą być zabezpieczone i ile te dane są warte. Dwa czynniki: okres ważności chronionych danych oraz wartość pieniężna tych danych umożliwiają nam obliczenie długości klucza, jaki musi być użyty, aby przez cały czas trwania ochrony tych danych nakład środków pieniężnych przeznaczonych na znalezienie właściwego klucza był większy niż wartość chronionych danych.

Dobre klucze są losowymi ciągami bitów. Jeżeli klucz ma 64 bity, to każdy możliwy klucz 64-bitowy musi być jednakowo prawdopodobny. Bity klucza generuje się korzystając ze źródła losowego lub z kryptograficznie bezpiecznego generatora bitów pseudolosowych.

7. Zakończenie

Wnioski wynikające z badań można zebrać w postaci następujących sformułowań:

1) im dłuższy plik, tym czas szyfrowania rośnie,
2) im dłuższy klucz, tym czas szyfrowania rośnie,
3) wzrost klucza nie wpływa proporcjonalnie na wzrost czasu szyfrowania (wpływa w mniejszym stopniu),
4) szybkość szyfrowania jest większa dla dłuższych plików (najmniej efektywne jest szyfrowanie małych plików),
5) różne metody charakteryzują się różnymi szybkościami pracy,
6) poszczególne tryby pracy w zasadniczy sposób nie wpływają na klasyfikację metod kryptograficznych w kontekście szybkości szyfrowania (występują czasami wahania, które generalnie nie mają znaczenia),
7) tryb ECB jest najszybszy.

W dalszym etapie prac dokonana zostanie próba formalnego ujęcia i wykorzystania tych wniosków.

LITERATURA

1. Borst J.: Two Attacks on Reduced IDEA. EUROCRYPT'97, pp. 1-13.
2. Daemen J.: Weak Keys for IDEA. CRYPTO'93, pp. 224-231.
3. ElGamal T.: A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. IEEE Transactions on Information Theory, v.IT-31, n.4, 1985, pp.469-472.
4. Elgamal T.: A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. Advances in Cryptology - CRYPTO '84 Procedings, Berlin: Springer-Verlag, 1985, pp.10-18.
5. Gilbert H., Chasse G.: A statistical attack of the FEAL-8 cryptosystem. CRYPTO'90, pp. 22-33.
6. Massey J. L.: SAFER K-64: A Byte-Oriented Block-Ciphering Algorithm. Fast Software Encryption nr 809. New York: Springer. 1994, pp. 1-17.
7. Matsui M., Yamagishi A.: A new method for known plaintext attack of FEAL cipher. EUROCRYPT'92, pp. 81-91.
8. Meier W.: On the Scurity of the IDEA Block Cipher. EUROCRYPT'93, pp. 371-385.
9. Miyaguchi S.: The FEAL Cipher Family. CRYPTO'90, pp. 627-638.
10. von Oorschort M.: Handbook of Applied Cryptography. Vanstone 1997.
11. Robling Denning D. E.: Kryptografia i ochrona danych. WNT, Warszawa 1992.
12. Schneier B.: Kryptografia dla praktyków. WILEY 1995.
13. Stokłosa J.: Kryptograficzna ochrona danych w systemach komputerowych. WNT, Warszawa 1994.
14. Stallings W.: Ochrona danych w sieci i intersieci. WNT, Warszawa 1997.

Recenzent: Dr inż. Ryszard Maceluch

Wpłynęło do Redakcji 24 marca 1999 r.

Abstract

In the paper author introduced results of the first stage of his work on the method of selection of coding algorithms for data protection. Author investigated some aspects of implementation of selected enciphering algorithms (FEAL, IDEA, SAFER), such as:

- Influence of the key size on coding time,
- Influence of the coded file size on coding time,
- Influence of the mode of work of encrypting algorithm on coding speed.

Using block coding, author used its implementation in different modes of work: ECB, CBC, CFB and OFM. Results received in the work one arranged in two groups:

- coding time for given algorithm in 4 modes of work,
- coding time for examined algorithms in each mode of work.

Results of speed examination one represented in three tables: for algorithm IDEA (Table 1), algorithm FEAL (Table 2) and for algorithm SAFER (Table 3).

For each mode (ECB, CBC, CFB, OFM) one represented time of coding of file and speed of coding. Obtained results permit on following conclusions:

- Speed of coding oscillated round definite values(85 MB/s),
- Speed of coding considerably comes down for smaller sizes of files,
- Among tested modes of work – fastest is ECB.

Results of tests one assembled for comprehensive comparison in Table 4. of Coding speed of different algorithms in each mode of work one pictured by means of graphs placed on drawings (Fig. 9, Fig. 10. Fig. 11, Fig. 12). On basis of these graphs is visible, that best speed among examined enciphering methods belongs to algorithm SAFER (about 30 kB/s in relation to FEAL)

Influence of the size of file on time of coding one examined also for ELGamal method (Fig. 13). Using this method, one also examined influence of key length on coding time (Table 5). From this is visible, that when key length expands then time of coding grows more or less in the same rate for very long keys the rate is even bigger.

In chapter 6 one thought about certain problems of analysis of encrypting codes (cryptoanalysis). However in the paper author restricted problem to brutal attack only, to show that from this point of view only, longer key gives much higher degree of safety (Fig. 14 and Fig. 15).