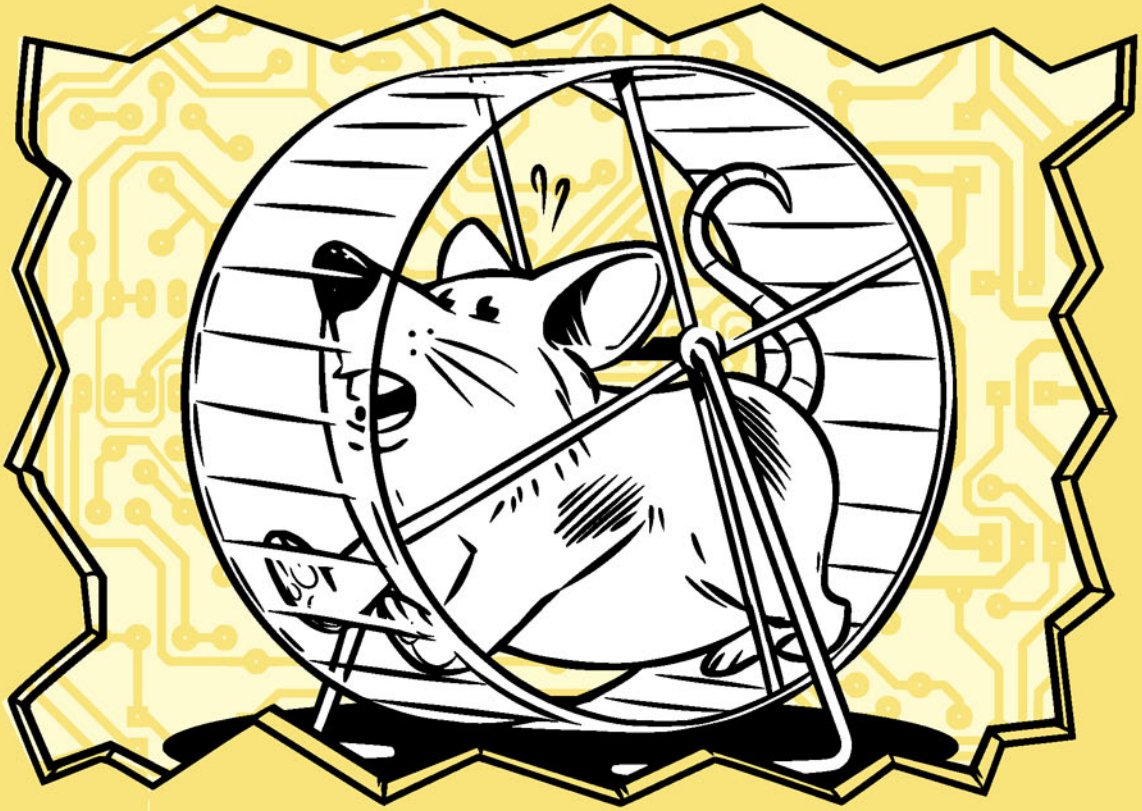
JAK DZIAŁA OPROGRAMOWANIE?

TAJEMNICE KOMPUTEROWYCH MECHANIZMÓW SZYFROWANIA, OBRAZOWANIA, WYSZUKIWANIA I INNYCH POWSZECHNIE UŻYWANYCH TECHNOLOGII

V. ANTON SPRAUL



no starch press

Helion

Tytuł oryginału: How Software Works: The Magic Behind Encryption, CGI, Search Engines, and Other Everyday Technologies

Tłumaczenie: Zdzisław Płoski

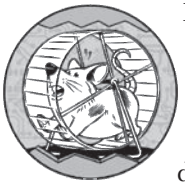
ISBN: 978-83-283-2593-7

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Spis treści

1

Szyfrowanie



LICZYMY, ŻE OPROGRAMOWANIE W KAŻDEJ CHWILI OCHRONI NASZE DANE, LECZ WIĘKSZOŚĆ Z NAS NIE BARDZO SIĘ ORIENTUJE, NA CZYM TA OCHRONA POLEGA. DLACZEGO IKONA „kłódki" w rogu Twojej przeglądarki oznacza, że możesz bezpiecznie wprowadzić numer swojej karty kredytowej? Jak to się dzieje, że opatrzenie Twojego telefonu hasłem rzeczywiście chroni zawarte w nim dane? Co w istocie zapobiega logowaniu się kogoś innego na Twoich kontach online?

Bezpieczeństwo komputerowe (ang. *computer security*) jest nauką o ochronie danych. W pewnym sensie bezpieczeństwo komputerowe reprezentuje technologię rozwiązującą problem spowodowany przez nią samą. Jeszcze nie tak dawno większość danych nie była przechowywana cyfrowo. Mieliśmy w biurach szafy z kartotekami, a pod łóżkami przechowywaliśmy pudełka po butach pełne fotografii. Oczywiście w tamtych czasach nie było łatwo podzielić się fotografiami z przyjaciółmi na świecie, nie mówiąc już o sprawdzeniu stanu konta na komórce, lecz nikt nie mógł ukraść Twoich prywatnych danych bez fizycznego ich zagarnięcia. Dziś nie tylko można Cię zdalnie obrabować, lecz możesz nawet nie wiedzieć, że do tej kradzieży doszło — przynajmniej do czasu otrzymania z banku zapytania, dlaczego wydajesz tysiące dolarów na karty podarunkowe[1].

[1] Ang. *gift cards*, rodzaj bonu towarowego o kształcie typowej karty bankomatowej — *przyp. tłum.*

W pierwszych trzech rozdziałach omówimy większość ważnych koncepcji leżących u podstaw bezpieczeństwa komputerowego. W tym rozdziale opowiemy o szyfrowaniu. Z natury rzeczy szyfrowanie umożliwia zamykanie danych w ten sposób, aby tylko *nam* wolno je było otworzyć. Omówione w następnych dwóch rozdziałach dodatkowe metody są potrzebne do zapewnienia pełnego repertuaru środków bezpieczeństwa, na którym opieramy swoje zaufanie, lecz szyfrowanie jest podstawą bezpieczeństwa komputerowego.

Cel szyfrowania

Pomyśl o pliku w swoim komputerze. Może on zawierać tekst, fotografię, arkusz elektroniczny, nagranie dźwiękowe lub wideo. Chcesz mieć dostęp do tego pliku, a jednocześnie trzymać go w sekrecie przed innymi. Tak przedstawia się podstawowy problem bezpieczeństwa komputerowego. W celu zachowania pliku w tajemnicy możesz się posłużyć **szyfrowaniem** (ang. *encryption*), aby nadać plikowi nowy format, nieczytelny dopóty, dopóki plik nie zostanie przywrócony do pierwotnej postaci za pomocą **deszyfrowania** (ang. *decryption*). Plik pierwotny jest zwany **tekstem jawnym** (ang. *plaintext*, nawet jeśli nie zawiera tekstu), a plik zaszyfrowany to **kryptogram** (ang. *ciphertext*)[2].

Atakujący (ang. *attacker*) to ktoś, kto próbuje odszyfrować kryptogram, nie będąc do tego upoważnionym. Celem szyfrowania jest utworzenie kryptogramu łatwego do odszyfrowania przez upoważnionych użytkowników, a jednocześnie praktycznie niemożliwego do odszyfrowania przez intruzów. Owo „praktycznie" przyprawia o ból głowy specjalistów od bezpieczeństwa. Tak jak żaden zamek nie jest nie do pokonania, nie ma szyfru absolutnie niemożliwego do odszyfrowania. Mając do dyspozycji dostatecznie dużo czasu i mocy obliczeniowej, teoretycznie można złamać każdy schemat szyfrowania. Celem bezpieczeństwa komputerowego jest utrudnienie roboty włamywaczowi w takim stopniu, żeby udane ataki były w praktyce niemożliwe, gdyż wymagałyby mocy obliczeniowych dla niego niedostępnych.

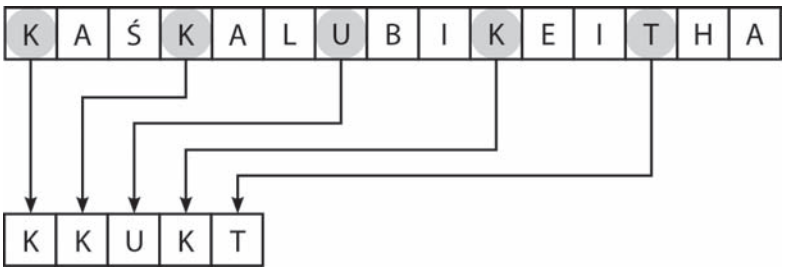
Zamiast skakać na główkę w odmęty szyfrowania opartego na oprogramowaniu, rozpocznę ten rozdział od prostych przykładów z czasów przedkomputerowych, gdy w grę wchodziły kody i zainteresowani nimi szpiedzy. Mimo że siła szyfrowania z biegiem lat niepomiernie wzrosła, u podstaw każdego szyfrowania leżą te same klasyczne sposoby. Później zobaczymy, jak te pomysły są wplecione w nowoczesny schemat szyfrowania cyfrowego.

[2] Tu i wielu innych miejscach stosujemy terminy zgodne z książką *Kryptografia* Douglasa R. Stinsona (WNT, Warszawa 2005, przekł. W. Bartol); kryptografia w obu językach ma ich w nadmiarze — *przyp. tłum.*

Przestawianie — te same dane, różny porządek

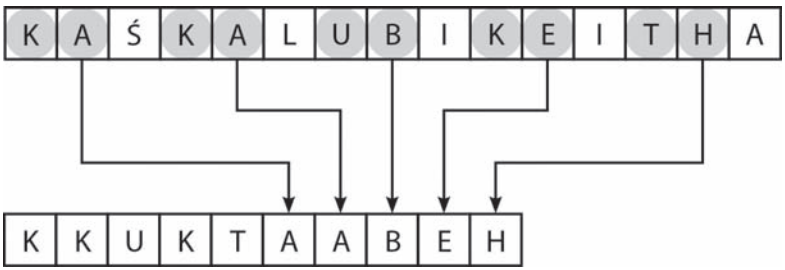
Jeden z najprostszych sposobów szyfrowania danych nazywa się **przestawianiem** (transpozycją, ang. *transposition*) i polega po prostu na „zmienianiu pozycji". Przestawianie jest rodzajem szyfrowania, którego moi koledzy i ja używaliśmy, przekazując sobie liściki w czasach szkolnych. Ponieważ te liściki przechodziły przez niepewne ręce, trzeba się było postarać, aby były niezrozumiałe dla wszystkich prócz nas.

Aby zapewnić naszym wiadomościom tajemnicę, zmienialiśmy w nich kolejność liter, używając prostego, łatwego do odwrócenia schematu. Przypuśćmy, że chciałem się podzielić jakże istotnym doniesieniem, że KAŚKA LUBI KEITHA (imiona zostały zmienione, gdyż moi rówieśnicy niczego tu nie zawinili). Żeby zaszyfrować tę wiadomość, skopiowałem każdą co trzecią literę tekstu jawnego (pomijając spacje). W rezultacie po pierwszym przejściu przez wiadomość skopiowałem pięć liter, co uwidacznia rysunek 1.1.



Rysunek 1.1. Pierwszy etap transpozycji przykładowej wiadomości

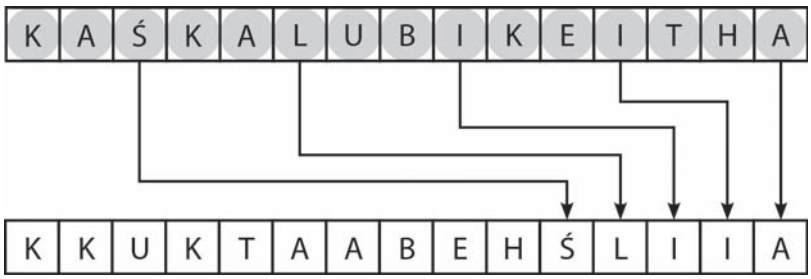
Po dotarciu do końca wiadomości wróciłem do początku i kontynuowałem wybieranie każdej co trzeciej litery z pozostałych. W drugim etapie otrzymałem stan przedstawiony na rysunku 1.2.



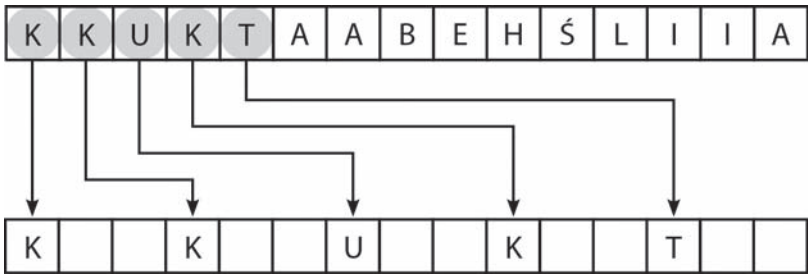
Rysunek 1.2. Drugi etap transpozycji

W ostatnim etapie skopiowałem pozostałe litery, co pokazano na rysunku 1.3.

Wynikowy kryptogram ma postać KKUKTAABEHŚLIIA. Moi koledzy mogli odczytać tę wiadomość, odwracając proces transpozycji. Pierwszy krok jest pokazany na rysunku 1.4. Przywrócenie wszystkich liter na ich pierwotne pozycje odkrywa tekst jawny.



Rysunek 1.3. Ostatni etap transpozycji



Rysunek 1.4. Pierwszy etap odwracania transpozycji w celu odszyfrowania

Ta elementarna metoda przestawień była zabawna w użyciu, lecz jest ona okropnie słabym szyfrowaniem. Największym problemem jest przeciek — jeden z moich kolegów wypaplał zasadę szyfrowania komuś spoza naszej paczki. Odtąd wysyłanie zaszyfrowanych wiadomości przestało być bezpieczne; wystarczyło tylko więcej popracować. Przecieków nie da się, niestety, uniknąć — i nie dotyczy to tylko dzieci w wieku szkolnym. Każda metoda jest podatna na przecieki, a im więcej ludzi korzysta z danej metody, tym większe prawdopodobieństwo, że zostanie wyjawiona.

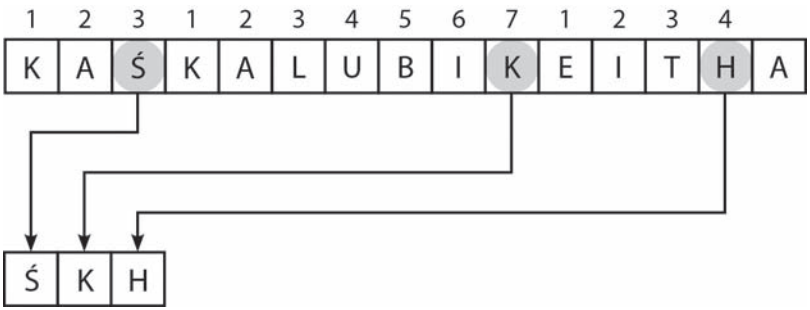
Z tego powodu wszystkie dobre systemy szyfrowania spełniają regułę określoną przez jednego z pierwszych kryptografów, Duńczyka Auguste'a Kerckhoffsa, znaną jako **zasada Kerckhoffsa**: bezpieczeństwo danych nie powinno zależeć od utrzymywania metody szyfrowania w tajemnicy.

Klucze szyfrów

Rodzi to oczywiste pytanie: w jaki sposób, jeśli metoda szyfrowania nie jest tajemnicą, możemy bezpiecznie szyfrować dane? Odpowiedź polega na następującej ogólnej i powszechnie znanej metodzie szyfrowania, w której jednak szyfrowanie poszczególnych wiadomości zmienia się przez zastosowanie **klucza szyfru** (ang. *cipher key*), czyli po prostu — **klucza**. Aby zrozumieć, czym jest klucz, przyjrzyjmy się ogólniejszej metodzie transpozycji.

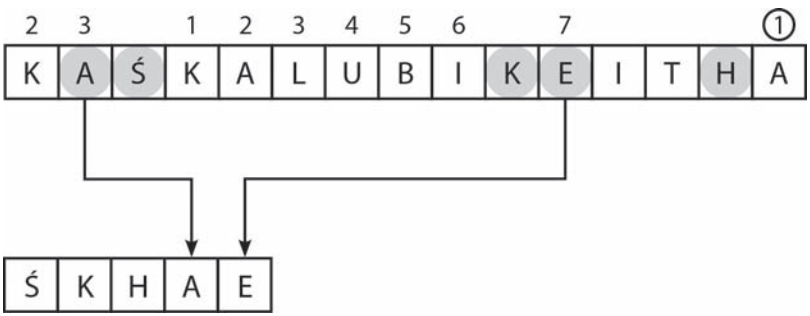
W tej metodzie nadawcy i odbiorcy, zanim zaczną przesyłać jakiekolwiek komunikaty, wchodzą w posiadanie wspólnej tajnej liczby. Powiedzmy, że moi koledzy i ja umówiliśmy się, że będzie to 374. Liczby tej użyjemy do zmiany

wzorca przestawiania w naszych kryptogramach, Dla wiadomości KAŚKA LUBI KEITHA wzorzec ten pokazano na rysunku 1.5. Cyfry naszej tajnej liczby wskazują, którą literę należy skopiować z tekstu jawnego do kryptogramu. Ponieważ pierwszą cyfrą jest 3, trzecia litera tekstu jawnego, czyli *Ś*, staje się pierwszą literą kryptogramu. Następną cyfrą jest 7, więc następną literą będzie siódma litera po *Ś*, czyli *K*. Dalej wybieramy czwartą literę, rozpoczynając odliczanie od następnej po *K*. Pierwszymi trzema literami kryptogramu będą *ŚKH*.



Rysunek 1.5. Pierwszy etap transpozycji z użyciem klucza 374

Na rysunku 1.6 pokazano, jak wygląda przekopiowanie dwóch następnych liter do kryptogramu. Zaczynając od miejsca, w którym przebywaliśmy (to jest od pozycji zaznaczonej na rysunku jedynką w kółku), odliczamy trzy pozycje, przy czym po dojściu do końca tekstu jawnego wracamy na jego początek, gdzie wybieramy *A* jako czwartą literę kryptogramu. Następnie wybieramy literę z siódmej pozycji po *A*, przeskakując litery już skopiowane — będzie nią *E*. Kontynuujemy to postępowanie, aż wszystkie litery z tekstu jawnego zostaną przestawione.



Rysunek 1.6. Drugi etap transpozycji z użyciem klucza 374

Tajna liczba 374 stała się zatem naszym kluczem szyfrowania. Gdyby ktoś przechwycił tę wiadomość, nie zdołałby jej zdeszyfrować bez tego klucza — nawet gdyby wiedział, że używamy metody przestawień. Klucz można regularnie zmieniać, aby ustrzec się przed ujawnieniem szyfrowania w wyniku niedyskrecji lub zdrady.

Łamanie szyfrów

Atakujący, nawet nie mając klucza, mogą próbować odtworzyć jawną postać tekstu innymi sposobami. Można zaatakować zaszyfrowane dane **metodą siłową** (ang. *brute force*), próbując użyć do kryptogramu metody szyfrowania wszystkimi możliwymi sposobami. W przypadku wiadomości zaszyfrowanej za pomocą przestawiania atak siłowy polegałby na sprawdzeniu wszystkich permutacji kryptogramu. Ponieważ atak siłowy jest prawie zawsze możliwy, liczba prób, którą atakujący będzie musiał wykonać, aby znaleźć tekst jawny, jest dobrym wskaźnikiem odporności szyfrowania. W naszym przykładzie wiadomość KAŚKA LUBI KEITHA ma około 40 miliardów permutacji.

Jest to wielka liczba, więc zamiast działania siłowego sprytny atakujący mógłby do szybszego odtworzenia tekstu jawnego posłużyć się odrobiną zdrowego rozsądku. Gdyby atakujący założył, że tekst jawny jest po polsku, większość permutacji można by wykluczyć jeszcze przed testowaniem. Na przykład atakujący mógłby założyć, że tekst jawny nie zaczyna się od grupy liter *HT*, ponieważ żadne polskie słowo nie zaczyna się od tej grupy liter[3]. To zaoszczędziłoby atakującemu sprawdzania miliarda permutacji.

Atakujący, który ma jakiś pomysł odnośnie do słów w wiadomości, może być jeszcze sprytniejszy w dochodzeniu do tekstu jawnego. W naszym przykładzie atakujący mógłby zgadnąć, że wiadomość zawiera imię kogoś z klasy. Można by sprawdzić, jakie imiona da się utworzyć z liter kryptogramu, a potem ustalić, jakie słowa można utworzyć z pozostałych liter.

Odgadnięte informacje dotyczące zawartości tekstu jawnego zwą się kryptoanalitycznymi ściągami lub krybami[4]. Najmocniejszym krybem jest **atak ze znanym tekstem jawnym** (ang. *known-plaintext attack*). Aby go dokonać, atakujący musi mieć dostęp do tekstu jawnego A, odpowiadającego A kryptogramu i kryptogramu B, w którym użyto tego samego klucza szyfrowania co w kryptogramie A. Okoliczności te, choć wyglądają na mało prawdopodobne, zdarzają się. Ludzie często zostawiają bez nadzoru dokumenty, co do których zakładają, że straciły status poufności, nie uświadamiając sobie, że mogą pomóc w atakach na inne dokumenty. Ataki ze znanym tekstem jawnym są silne; odkrycie wzorca transpozycji jest łatwe, gdy masz przed oczami zarówno tekst jawny, jak i kryptogram.

Najlepszą obroną przed atakami ze znanym tekstem jawnym są dobre praktyki bezpieczeństwa, takie jak regularne zmienianie haseł. Niemniej nawet przestrzeganie najlepszych zasad bezpieczeństwa nie zapobiegnie temu, że atakujący zawsze będą mieli pewne wyobrażenie o zawartości tekstu jawnego (to dlatego właśnie są tak zainteresowani jego odczytaniem). W wielu przypadkach znają

[3] W oryginale jest oczywiście mowa o języku angielskim, ale również w języku polskim żadne słowo nie zaczyna się od „ht" — *przyp. tłum.*

[4] Punktami zaczepienia, wskazówkami, ściągami (ang. *crib* — dosł. łóżeczko, żłóbek, oszalowanie, ściąga, bryk); nazwa używana przez kryptologów z czasów Enigmy, zostawiamy ją w transkrypcji fonetycznej, bez przypisania — *przyp. tłum.*

oni większość tekstu jawnego i mogą mieć dostęp do znanych par tekst jawny – kryptogram. Dobry system kryptograficzny powinien sprawiać, że kryby i znane teksty jawne staną się dla atakujących bezużyteczne.

Podstawianie — zastępowanie danych

Drugi podstawowy sposób szyfrowania jest odporniejszy na kryby. Zamiast przesuwania danych w **metodach podstawieniowych** (ang. *substitution methods*) poszczególne fragmenty danych są systematycznie zastępowane innymi. W przypadku komunikatów tekstowych najprostszą postacią podstawiania jest zamiana każdego wystąpienia jednej litery na inną. Na przykład każde *A* staje się *D*, każde *B* — *H* itd. Klucz tego typu szyfrowania wygląda tak, jak w tabeli 1.1.

Tabela 1.1. Klucz szyfrowania podstawieniowego

Oryginał	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
Zastąpienie	M	N	B	V	C	X	Z	L	K	F	H	G	J	D	S	A	P	O	I	U	Y	T	R	E	W	Q

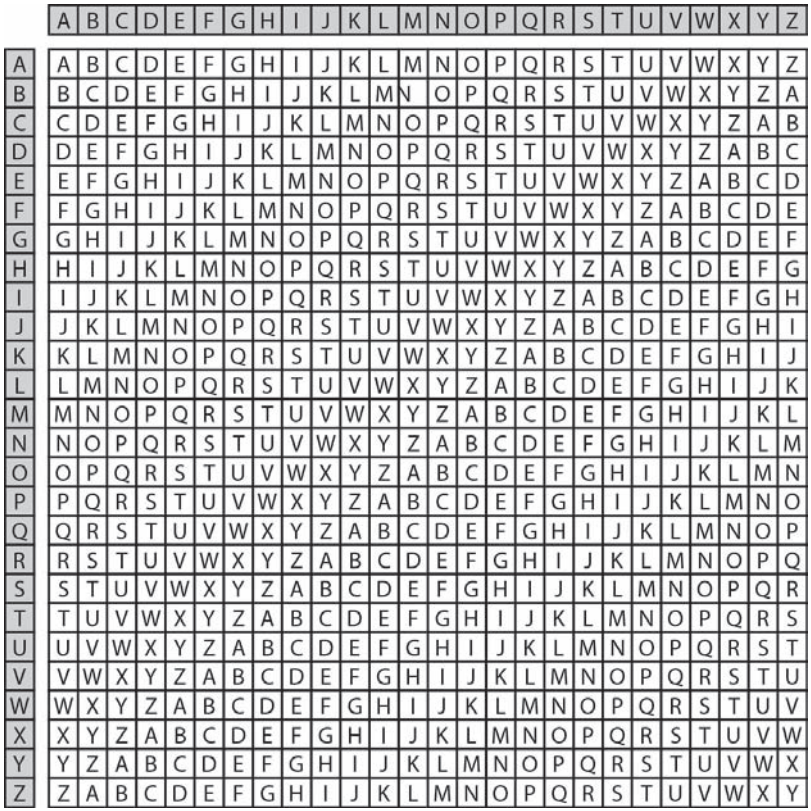
Choć **prosty szyfr podstawieniowy**, jak zwą tę metodę, jest ulepszeniem transpozycji, również przysparza problemów. Podstawień jest niezbyt wiele, więc atakujący może niekiedy odszyfrować kryptogram metodą siłową.

Proste podstawienie jest również podatne na **analizę frekwencyjną** (ang. *frequency analysis*), w której atakujący wykorzystuje wiedzę o częstości występowania liter lub ich kombinacji w danym języku. Ujmując ogólnie: znajomość częstości występowania w tekście jawnym jednostek danych daje atakującemu przewagę. Na przykład litera *E* występuje w tekstach angielskich najczęściej, a *TH* jest najpopularniejszą w tym języku parą liter. Dlatego litera najczęściej występująca w długim kryptogramie będzie prawdopodobnie reprezentowała literę *E*, a najczęściej występująca para liter będzie odpowiadała *TH*.

Siła analizy frekwencyjnej polega na tym, że szyfr podstawieniowy staje się coraz bardziej podatny na złamanie wraz ze zwiększaniem objętości tekstu. Ataki są również łatwiejsze po zgromadzeniu zbioru kryptogramów zaszyfrowanych tym samym kluczem; unikanie takiego **ponownego użycia klucza** (ang. *key reuse*) jest ważnym zaleceniem z punktu widzenia bezpieczeństwa.

Zmienianie wzorca podstawiania

Aby uodpornić szyfrowanie na analizę frekwencyjną, możemy zmieniać wzorzec podstawiania w trakcie szyfrowania. Tak więc pierwsze *E* w tekście jawnym mogłoby być zastąpione przez *A*, lecz drugie *E* w tekście jawnym można by już zastąpić przez *T*. Ten sposób jest określany jako **podstawianie wieloalfabetowe** (ang. *polyalphabetic substitution*). W jednej z metod podstawiania wieloalfabetowego korzysta się z siatki alfabetów zwanej **tabula recta**, pokazanej na rysunku 1.7. W tej



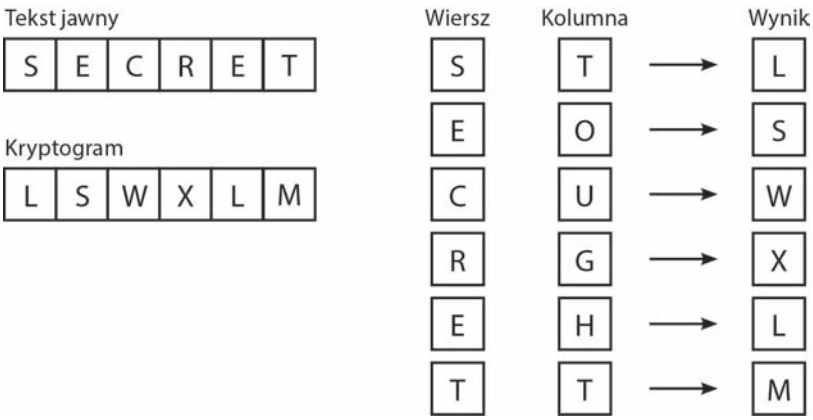
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	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
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
Y	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
Z	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

Rysunek 1.7. Tabula recta (zacieniowane pierwsza kolumna i pierwszy wiersz zawierają etykiety)

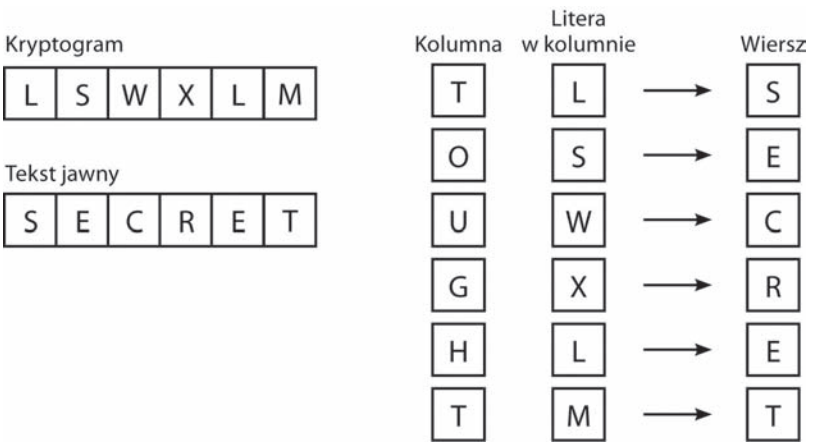
tabeli każdy wiersz i kolumna są poetykietowane literą alfabetu, która rozpoczyna dany wiersz lub kolumnę. Każde oczko w siatce jest lokalizowane za pomocą dwóch liter, na przykład w wierszu D i kolumnie H występuje litera *K*.

Z zastosowaniem tabuli recty klucz ma postać tekstową: zamiast liczb, których używaliśmy w przykładzie z przestawianiem, do zmiany szyfrowania są tutaj używane litery. Litery tekstu jawnego identyfikują wiersze w tabuli rekcie, a litery klucza wskazują kolumny. Załóżmy na przykład, że tekst jawny ma postać *SECRET*, a naszym kluczem szyfrowania jest słowo *TOUGH*. Ponieważ pierwszą literą tekstu jawnego jest *S*, a pierwszą literą klucza jest *T*, pierwsza litera kryptogramu znajduje się w tabuli rekcie w wierszu S i kolumnie T — jest nią litera *L*. Następnie posługujemy się kolumną O tabeli do zaszyfrowania drugiej litery tekstu jawnego, to jest *E* (w wyniku otrzymujemy *S*) itd., jak pokazano na rysunku 1.8. Ponieważ tekst jawny jest dłuższy niż klucz, musimy ponownie użyć pierwszej litery klucza.

Deszyfrowanie polega na odwróceniu tego postępowania, co pokazano na rysunku 1.9. Litery klucza wskazują kolumny przeglądane w celu znalezienia odpowiedniej litery w kryptogramie. Wiersz z odnalezioną literą kryptogramu wskazuje literę tekstu jawnego. W naszym przykładzie pierwszą literą klucza jest *T*,



Rysunek 1.8. Szyfrowanie z użyciem tabuli recty i klucza szyfru TOUGH



Rysunek 1.9. Deszyfrowanie z użyciem tabuli recty i klucza szyfru TOUGH

a pierwszą literą kryptogramu jest *L*. Przeglądamy kolumnę T tabuli recty w poszukiwaniu *L*; ponieważ *L* występuje w wierszu S, literą tekstu jawnego jest *S*. Proces powtarza się dla każdej litery kryptogramu.

Podstawianie wieloalfabetowe jest efektywniejsze niż proste podstawianie, gdyż w trakcie szyfrowania komunikatu zmienia się w nim wzorzec podstawiania. W naszym przykładzie dwa wystąpienia *E* w tekście jawnym przybrały postaci różnych liter, a dwa wystąpienia *L* w kryptogramie reprezentują różne litery tekstu jawnego.

Poszerzanie klucza

Choć podstawianie wieloalfabetowe jest znacznym ulepszeniem prostego szyfru podstawieniowego, skutkuje tylko wówczas, gdy klucz nie jest zbyt często powtarzany. W przeciwnym razie występują w nim te same problemy co w prostym podstawianiu. Na przykład w przypadku klucza o długości 5 każda litera tekstu

jawnego byłaby reprezentowana tylko przez pięć różnych liter w kryptogramie, co wystawiałoby długie kryptogramy na niebezpieczeństwo analizy frekwencyjnej i kryby. Atakujący musiałby się więcej napracować, lecz mając do czynienia z dostatecznie dużą ilością tekstu zaszyfrowanego, nadal mógłby złamać szyfr.

W celu maksymalizacji efektywności musimy używać kluczy szyfrowania tak długich jak tekst jawny. Tę technikę nazywa się **podkładką jednorazową** (ang. *one-time pad*). W większości sytuacji nie jest to jednak rozwiązanie praktyczne. Zamiast tego metoda zwana **poszerzaniem klucza** (ang. *key expansion*) umożliwia osiąganie efektu użycia dłuższych kluczy za pomocą krótkich kluczy. Jedno z odzwierciedleń tego pomysłu często pojawia się w powieściach szpiegowskich. Zamiast współużytkowania superdługiego klucza dwóch szpiegów chcących wymieniać meldunki uzgadnia **książkę kodową** (ang. *code book*) służącą jako magazyn długich kluczy. Aby uniknąć podejrzeń, książka kodowa jest zwykłym egzemplarzem jakiejś literatury, na przykład specjalnym wydaniem sztuk Szekspira.

Załóżmy, że według tego schematu ma być przesłany 50-literowy komunikat. Oprócz kryptogramu nadawca dołącza również nierozwinięty klucz. Z użyciem dzieł Szekspira jako książki kodowej nierozwinięty klucz może przyjąć postać 2.2.4.9. Pierwsza dwójka wskazuje drugą sztukę Szekspira w układzie alfabetycznym (*As You Like It*[5]). Druga dwójka oznacza akt II tej sztuki. Czwórka oznacza czwartą scenę tego aktu. Dziewiątka symbolizuje dziewiąte zdanie tej sceny w określonym wydaniu: „When I was at home, I was in a better place, but travelers must be content"[6]. Liczba liter w tym zdaniu przekracza liczbę liter w tekście jawnym, można więc jej użyć do szyfrowania i deszyfrowania za pomocą tabuli recty — jak poprzednio. W ten sposób stosunkowo krótki klucz może być poszerzony tak, aby pasował do konkretnego komunikatu.

Zauważmy, że tego schematu nie kwalifikujemy jako podkładki jednorazowej, gdyż książka kodowa jest skończona, dlatego zdań kluczy trzeba będzie w końcu użyć powtórnie. Oznacza to jednak, że nasi szpiedzy muszą pamiętać tylko krótkie klucze cyfrowe, bezpieczniej szyfrując swoje meldunki dłuższymi kluczami. Jak się przekonamy, pomysł poszerzania klucza jest ważny w szyfrowaniu komputerowym, ponieważ wymagane klucze szyfrowania są wielkie, lecz muszą być pamiętane w mniejszych postaciach.

Zaawansowany standard szyfrowania

Skoro zobaczyliśmy już z osobna, jak działa transpozycja, podstawianie i poszerzanie klucza, przyjrzyjmy się, jak ze starannego połączenia wszystkich tych trzech technik uzyskuje się bezpieczne szyfrowanie cyfrowe.

---

[5] Czyli *Jak wam się podoba — przyp. tłum.*

[6] Z ang.: „[...] kiedym był w domu, lepiej mi było; ale podróżny musi przestać na małym" (przekł. L. Urlich. W. Szekspir, *Dzieła dramatyczne*, tom II, *Komedie*. PIW, Warszawa 1958) — *przyp. tłum.*

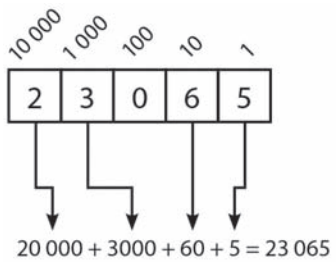
Zaawansowany standard szyfrowania (*AES*, ang. *Advanced Encryption Standard*) jest standardem otwartym, co oznacza, że jego specyfikacja może być urzeczywistniona przez każdego bez opłacania licencji. Czy sobie z tego zdajesz sprawę, czy nie, większość Twoich danych jest chroniona z użyciem AES. Jeśli w domu lub w pracy masz bezpieczną sieć bezprzewodową, jeśli kiedykolwiek zdarzyło Ci się zabezpieczać hasłem plik w archiwum *.zip* lub jeśli używasz karty kredytowej w sklepie albo pobierasz pieniądze z bankomatu — prawdopodobnie polegasz (przynajmniej częściowo) na AES.

Podstawy dwójkowe

Jak dotąd uciekałem się do przykładów szyfrowania tekstów w trosce o ich prostotę. Jednakże dane szyfrowane przez komputery są przedstawiane w postaci liczb dwójkowych. Jeśli wcześniej nie miałeś z nimi do czynienia, tutaj przedstawiam mały wstęp.

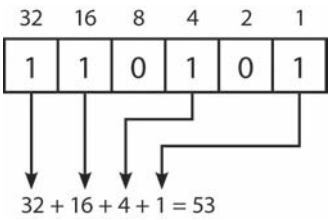
System dziesiętny a system dwójkowy

System liczbowy, do którego wszyscy nawykliśmy od dziecka, jest nazywany **dziesiętnym** (ang. *decimal*, „deci" z łaciny znaczy „dziesięć"), gdyż używa się w nim dziesięciu cyfr, od 0 do 9. Każda cyfra w zapisie liczby reprezentuje liczbę jednostek 10-krotnie większą niż cyfra sąsiadująca z nią z prawej strony. Jednostki i wielkości określane liczbą 23 065 pokazano na rysunku 1.10. Na przykład dwójka na piątej pozycji po lewej oznacza, że mamy dwie „dziesiątki tysięcy", a szóstka oznacza sześć „dziesiątek".



Rysunek 1.10. Każda cyfra w zapisie pozycyjnym liczby dziesiętnej 23 065 reprezentuje inną liczbę jedności

W **dwójkowym** (binarnym, ang. *binary*) systemie liczbowym istnieją tylko dwie cyfry: 0 i 1, nazywane **bitami** od angielskiego ***bi**nary digi**ts*** (cyfry dwójkowe). Każdy bit w liczbie dwójkowej reprezentuje wielkość dwa razy większą niż bit po prawej. Jednostki i wielkości w liczbie dwójkowej 110101 pokazano na rysunku 1.11. Jak widać, mamy tu po jednej z następujących wielkości: 32, 16, 4 i 1. Wobec tego liczba dwójkowa reprezentuje sumę tych czterech wartości, co daje dziesiętnie liczbę 53.



Rysunek 1.11. Każda cyfra w zapisie pozycyjnym liczby dwójkowej 110101 reprezentuje inną liczbę jedności

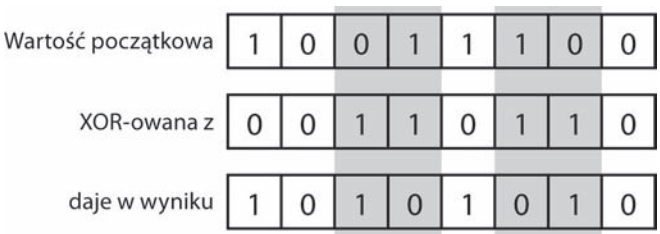
Liczby dwójkowe są zazwyczaj zapisywane z użyciem stałej liczby bitów. Najpopularniejszą długością liczby dwójkowej jest osiem bitów określanych jako **bajt** (ang. *byte*). Choć dziesiętną liczbę 53 można zapisać dwójkowo w postaci

110101, zapisanie jej w postaci bajta wymaga ośmiu bitów, toteż początkowe bity wypełnia się zerami — otrzymujemy więc zapis 00110101. Bajt o najmniejszej wartości, 00000000, reprezentuje dziesiętne 0; największy możliwy bajt, 11111111, przedstawia liczbę 255.

Operacje na bitach

Oprócz zwykłych operacji matematycznych, jak dodawanie i mnożenie, oprogramowanie używa pewnych operacji charakterystycznych dla liczb dwójkowych. Nazywa się je **operacjami bitowymi** (ang. *bitwise operations*), ponieważ są wykonywane z osobna na każdym bicie, a nie na liczbie dwójkowej traktowanej jako całość.

W szyfrowaniu powszechnie używa się operacji bitowej zwanej **alternatywą wykluczającą** (ang. *exclusive-or*, *XOR*). Podczas wykonywania na dwóch liczbach dwójkowych operacji XOR jedynki drugiej liczby przełączają[7] wartości odpowiednich bitów pierwszej liczby, jak pokazano na rysunku 1.12.



Rysunek 1.12. Operacja alternatywy wykluczającej (XOR). Bity 1 w drugim bajcie wskazują, które bity są „przełączane" w pierwszym bajcie, co pokazano w zacieniowanych kolumnach

Pamiętamy, że szyfrowanie musi być odwracalne. XOR zmienia wzorce bitowe w sposób niemożliwy do przewidzenia bez znajomości użytych liczb dwójkowych, lecz jest to łatwe do odwrócenia. XOR-owanie wyniku za pomocą drugiej liczby przełącza te same bity do ich pierwotnego stanu, jak pokazano na rysunku 1.13.

Zamiana danych na postać dwójkową

Komputery używają liczb dwójkowych do reprezentowania wszelkich danych. Plik z tekstem jawnym mógłby być wiadomością tekstową, arkuszem kalkulacyjnym, obrazem, wideoplikiem lub czymkolwiek innym, lecz koniec końców każdy plik jest ciągiem bajtów. Większość danych komputerowych od początku ma postać numeryczną, może więc być bezpośrednio zamieniona na liczby dwójkowe. Jednak w pewnych przypadkach trzeba zastosować specjalny system kodowania do zamiany danych nienumerycznych na postać dwójkową.

[7] To znaczy zmieniają na przeciwne — *przyp. tłum.*

Rysunek 1.13. Jeśli potraktujemy dwukrotnie bajt operacją XOR z użyciem tego samego bajta, otrzymamy to, co mieliśmy na początku

Aby zobaczyć, jak komunikat tekstowy staje się ciągiem bajtów, rozważmy wiadomość:

```
Send more money!
```
[8]

Ta wiadomość składa się z 16 znaków, licząc litery, spacje i wykrzyknik. Każdy znak możemy wyrazić za pomocą bajta, korzystając na przykład z systemu o nazwie *American Standard Code for Information Interchange*[9], znanego również pod postacią skrótu *ASCII* (wym. „aski"), W systemie ASCII duża litera *A* jest reprezentowana liczbą 65, *B* ma przypisaną wartość 66 itd. aż do 90 dla Z. W tabeli 1.2 pokazano garść wybranych pozycji z tabeli (kodu) ASCII.

Szyfrowanie AES w ujęciu ogólnym

Zanim przyjrzymy się szczegółom szyfrowania AES, omówimy ten proces pobieżnie.

Klucze szyfrowania w AES są liczbami dwójkowymi. Długość klucza może się zmieniać, lecz omówimy najprostszą wersję AES, w której jest stosowany klucz 128-bitowy. Za pomocą matematycznego poszerzania klucza AES przekształca pierwotny 128-bitowy klucz w jedenaście kluczy 128-bitowych.

Tekst jawny jest dzielony w AES na bloki po 16 bajtów w siatce 4×4; siatkę dla przykładowego komunikatu *Send more money!* przedstawiono na rysunku 1.14. Grube linie oddzielają szesnaście bajtów, a cienkie linie oddzielają bity w bajtach.

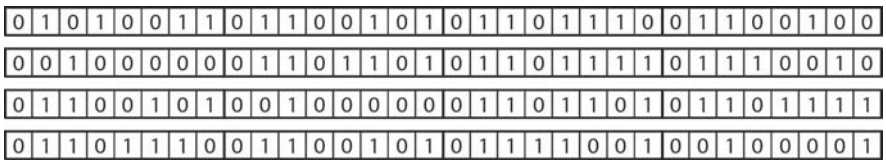
Dane tekstu jawnego są dzielone na tyle 16-bajtowych bloków, ile trzeba. Jeśli ostatni blok nie jest pełny, jego resztę uzupełnia się losowymi liczbami dwójkowymi.

[8] Z ang. przyślijcie więcej pieniędzy! — *przyp. tłum.*

[9] Z ang. standardowy amerykański kod wymiany informacji — *przyp. tłum.*

Tabela 1.2. Wybrane pozycje z tabeli ASCII

Znak	Liczba dziesiętna	Bajt dwójkowy
(spacja)	32	01000000
!	33	01000001
,	44	00101100
.	46	00101110
A	65	01000001
B	66	01000010
C	67	01000011
D	68	01000100
E	69	01000101
a	97	01100001
b	98	01100010
c	99	01100011
d	100	01100100
e	101	01100101



Rysunek 1.14. Przykładowy komunikat „Send more money!" przekształcony na siatkę bajtów, gotowy do szyfrowania metodą AES

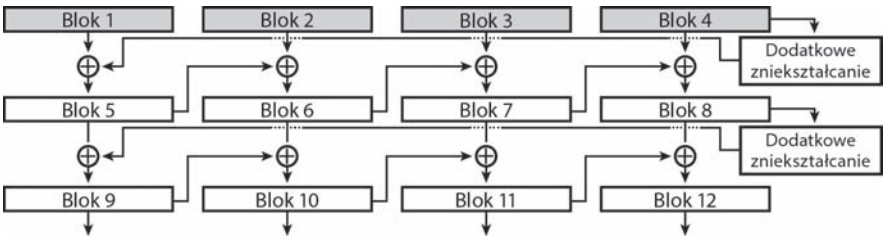
AES poddaje następnie każdy 16-bajtowy blok 10 **rundom** (ang. *rounds*) szyfrowania. W jednej rundzie bajty są przestawiane wewnątrz bloku i podstawiane z użyciem tabeli. Potem za pomocą operacji XOR bajty bloku są łączone ze sobą i z jednym ze 128-bitowych kluczy.

Tak wygląda szyfr AES w największym skrócie. Przyjrzyjmy się teraz jego kilku krokom bardziej szczegółowo.

Poszerzanie klucza w AES

Poszerzanie klucza w systemie szyfrowania cyfrowego różni się nieco od omówionego przez nas wcześniej z „książką kodową". Zamiast po prostu zaglądać po dłuższy klucz do książki, AES poszerza klucz za pomocą tych samych narzędzi, których później użyje do samego szyfrowania: operacji XOR, transpozycji i prostego podstawiania.

Rysunek 1.15 przedstawia kilka pierwszych faz procesu poszerzania klucza. Każdy blok na rysunku ma 32 bity, a jeden wiersz na tym rysunku przedstawia jeden klucz 128-bitowy. Oryginalny 128-bitowy klucz składa się z pierwszych czte-



Rysunek 1.15. Proces poszerzania klucza w AES

rech bloków zacieniowanych na rysunku. Każdy inny blok jest wynikiem operacji XOR na dwu poprzednich blokach; operacja XOR jest przedstawiona za pomocą znaku plus w kółku. Na przykład blok 6 powstaje z wykonania XOR na blokach 2 i 5.

Jak widać na rysunku, po prawej stronie, co czwarty blok jest poddawany zabiegowi oznaczonemu etykietą „dodatkowe zniekształcanie". Ten proces obejmuje przestawianie bajtów w bloku i podstawianie w każdym bajcie wartości zgodnie z tablicą o nazwie **S-boks** (ang. *S-box*).

Tablica S-boks, używana zarówno w poszerzaniu klucza, jak i później, w samym szyfrowaniu, jest starannie zaprojektowana z myślą o wzmocnieniu różnic w tekście jawnym. Oznacza to, że dwa podobne do siebie bajty tekstu jawnego będą miały z reguły zupełnie różne podstawienia w S-boksie. Pierwszych osiem pozycji tej tabeli pokazano w tabeli 1.3.

Tabela 1.3. Wyjątki z tablicy S-boks

Oryginalny wzorzec bitowy	Zastępujący wzorzec bitowy
00000000	01100011
00000001	01111100
00000010	01110111
00000011	01111011
00000100	11110010
00000101	01101111
00000110	01101111
00000111	11000101
00001000	00110000
00001001	00000001

Rundy szyfrowania AES

Kiedy AES utworzy wszystkie potrzebne klucze, można rozpocząć faktyczne szyfrowanie. Przypomnijmy, że binarny tekst jawny jest pamiętany w siatce 16 bajtów, czyli na 128 bitach, to znaczy ma taką samą długość jak pierwotny klucz. Nie jest to przypadek. Pierwszy krok rzeczywistego szyfrowania polega na wykonaniu XOR

128-bitowej siatki danych z pierwotnym 128-bitowym kluczem. Teraz robota rusza na dobre, jako że siatka danych jest poddawana 10 rundom cyfrowego zniekształcania. Każda runda składa się z czterech kroków.

1. **Podstawianie**

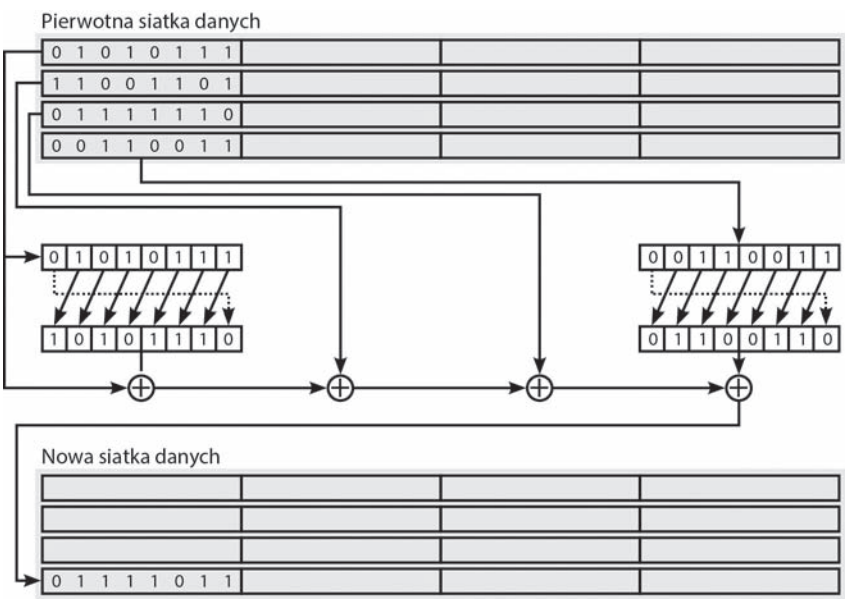
   Każdy z 16 bajtów w siatce jest zastępowany z użyciem tej samej tablicy S-boks, która była użyta do poszerzania klucza.

2. **Transpozycja wiersza**

   Następnie bajty są przesuwane na różne pozycje w obrębie swojego wiersza w siatce.

3. **Kombinacja kolumn**

   Dalej dla każdego bajta w siatce jest obliczany nowy bajt na podstawie kombinacji wszystkich czterech bajtów w danej kolumnie. W tym obliczeniu znów występuje operacja XOR, a także binarna odmiana przestawiania. Aby dać posmak tego procesu, na rysunku 1.16 ukazujemy obliczenie skrajnego lewego bajta w najniższym wierszu. Cztery bajty skrajnej lewej kolumny są XOR-owane razem po uprzednim przestawieniu bitów w górnym i dolnym bajcie kolumny. Ten rodzaj przestawienia nosi nazwę **rotacji bitowej** (ang. *bitwise rotation*)[10]: bity są przesuwane o jedną pozycję w lewo, a „wypadający" skrajny bit z lewej trafia na skrajną pozycję po prawej stronie.



Rysunek 1.16. Jedna część kroku zniekształcania kolumnowego w rundzie AES

[10] Inna nazwa: przesuwanie cykliczne — *przyp. tłum.*

Każdy bajt w nowej siatce jest obliczany podobnie, przez kombinację bajtów w kolumnie za pomocą XOR; jedyna różnica dotyczy wyboru bajtów, których bity są rotowane przed wykonaniem XOR.

4. **Wykonanie XOR z kluczem szyfru**

   Na koniec siatka powstała w poprzednim kroku jest XOR-owana z kluczem tej rundy. Dlatego właśnie trzeba poszerzyć klucz, aby w każdej rundzie operacja XOR była wykonywana z innym kluczem.

Podczas deszyfrowania AES są wykonywane te same kroki, co podczas szyfrowania, tylko w odwrotnej kolejności. Ponieważ jedynymi operacjami w szyfrowaniu są XOR-y, proste podstawianie z S-boksu i przestawienia bitów i bajtów, wszystko jest odwracalne, jeśli klucz jest znany.

Łańcuchowanie bloków

Szyfrowanie AES można by stosować z osobna do każdego 16-bajtowego bloku pliku, lecz narażałoby to kryptogram na niebezpieczeństwa. Jak powiedzieliśmy, im częściej jest używany klucz szyfrowania, tym większa możliwość, że atakujący odkryją i wykorzystają wzorce. Pliki komputerowe są niejednokrotnie olbrzymie, więc używanie tego samego klucza do szyfrowania milionów bloków jest formą ponownego użytkowania klucza na wielką skalę, co sprawia, że kryptogram może być poddany analizie frekwencyjnej i podobnym zabiegom.

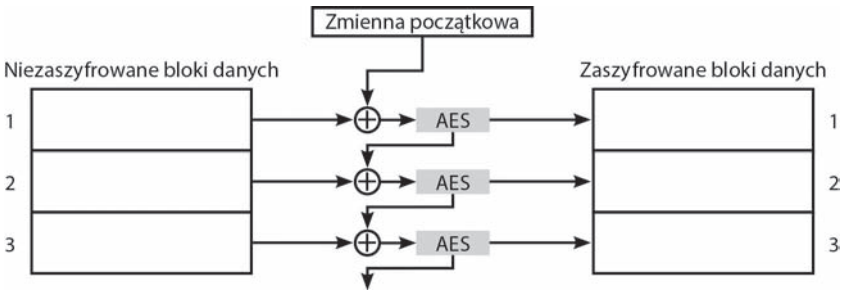
Z tego powodu systemy szyfrowania blokowego, takie jak AES, są modyfikowane, aby z identycznych bloków w tekście jawnym powstawały inne bloki w kryptogramie. Jedna z takich modyfikacji zwie się **łańcuchowaniem bloków** (ang. *block chaining*).

Podczas łańcuchowania bloków pierwszy blok tekstu jawnego jest przed zaszyfrowaniem XOR-owany z losową 128-bitową liczbą. Ta losowa liczba nazywa się **zmienną początkową** (ang. *starting variable*) i jest przechowywana wraz z kryptogramem. Ponieważ każdemu szyfrowaniu jest przyporządkowana losowa zmienna początkowa, dwa pliki zaczynające się tymi samymi danymi będą miały inne kryptogramy po zaszyfrowaniu — nawet wówczas, gdy do szyfrowania użyje się tego samego klucza.

Każdy następny blok tekstu jawnego jest przed szyfrowaniem XOR-owany z poprzednim blokiem kryptogramu, „łańcuchując" szyfrowanie, jak pokazano na rysunku 1.17. Łańcuchowanie zapewnia, że dublowane bloki w tekście jawnym będą się różniły w kryptogramie. To oznacza, że dowolnej długości pliki można szyfrować bez obawy o analizę frekwencyjną.

Dlaczego AES jest bezpieczny

Jak widzieliśmy, choć AES składa się z wielu kroków, każdy z nich jest transpozycją lub prostym podstawieniem. Dlaczego AES jest uważany za dość silny, żeby chronić dane świata? Pamiętajmy: atakujący używają metody siłowej lub krybów, lub wykorzystują wzorce w kryptogramie. AES ma świetną obronę przed wszystkimi tymi metodami ataku.



Rysunek 1.17. Szyfrowanie AES z łańcuchowaniem bloków

W AES działanie siłowe oznacza poddawanie kryptogramu procesowi deszyfrowania dla każdego możliwego klucza, aż do wytworzenia tekstu jawnego. Klucze w AES mają 128, 192 lub 256 bitów. Nawet najmniejszy klucz pozwala utworzyć około 300 000 000 000 000 000 000 000 000 000 000 000 000 możliwych kluczy, a w ataku siłowym, aby móc oczekiwać trafienia na właściwy, należałoby uwzględnić około połowy z nich. Atakujący z komputerem mogącym wypróbowywać milion kluczy na sekundę zdołałby wypróbować dziennie 1 000 000 × 60 (sekund) × 60 (minut) × 24 (godziny) = 86 400 000 000 kluczy. Przez rok atakujący zdołałby wypróbować 31 536 000 000 000 kluczy. Mimo że jest to liczba ogromna, nie stanowi nawet miliardowej części jednej miliardowej liczby możliwych kombinacji. Atakujący mógłby zaopatrzyć się w większą moc obliczeniową, lecz nawet wówczas wypróbowanie tak wielu kluczy nie wydaje się osiągalne, przy czym dotyczy to zaledwie wersji 128-bitowej.

AES utrudnia również używanie krybów lub znajdowanie wzorców przydatnych do wykorzystania. W każdej rundzie szyfrowania AES rotuje bajty w każdym wierszu siatki i tworzy kombinacje bajtów w każdej kolumnie. Po wielu takich rundach bajty są starannie wymieszane, toteż finalna wartość każdego z nich w siatce kryptogramu zależy od początkowych wartości wszystkich bajtów w siatce tekstu jawnego. Ta cecha szyfrowania jest nazywana **dyfuzją** (przenikaniem).

Ponadto przekazywanie bajtów runda po rundzie za pośrednictwem S-boksu wzmacnia efekt dyfuzji, a łańcuchowanie bloków przenosi efekty dyfuzji każdego bloku do bloku następnego. Wszystko razem wytwarza w szyfrowaniu AES **efekt lawiny** (ang. *avalanche property*), w którym małe zmiany w tekście jawnym powodują wszechogarniające zmiany w kryptogramie.

AES krzyżuje atakującym szyki niezależnie od tego, ile wiedzą o ogólnym wyglądzie tekstu jawnego. Na przykład jakaś firma mogłaby wysyłać klientom e-maile, wykorzystując wspólny szablon, w którym zmieniają się tylko numery kont klientów i niezapłacone należności. Dzięki dyfuzji, lawinowości i łańcuchowaniu bloków zaszyfrowane teksty tych listeli będą się bardzo różniły. Dyfuzja i lawinowość redukuje również liczbę wzorców, które można by wykorzystać do analizy frekwencyjnej. Nawet ogromny plik z tekstem jawnym, składający się z wielokrotnie powtórzonego tego samego, 16-bajtowego bloku, zostanie w wyniku zaszyfrowania przez AES z łańcuchowaniem bloków przeobrażony w wyglądającą losowo gmatwaninę bitów.

Możliwe ataki na AES

Szyfr AES okazuje się odporny na konwencjonalne ataki, lecz czy skrywa jakieś słabości, które umożliwiałyby drogę na skróty w dążeniu do znalezienia właściwego klucza szyfrowania? Nie ma jasnej odpowiedzi, ponieważ udowodnienie wady jest trudne. Stwierdzenie, że nie są znane drogi na skróty, czyli możliwości krakerskie (ang. *cracks*, szybkie złamanie szyfru), to jedno. Czym innym jest udowodnienie, że *nie mogą* istnieć. Kryptografia jest nauką, a nauka (poznanie) zawsze przesuwa swoje granice. Po prostu nie znamy kryptografii i jej metod matematycznych w takim stopniu, aby przesądzać, co jest niemożliwe.

Część utrudnień w analizie podatności na ataki otwartego standardu, takiego jak AES, wynika stąd, że programiści implementujący standard, to jest kodujący go, mogą mimowolnie wprowadzić jakieś braki w jego bezpieczeństwie. Na przykład niektóre implementacje AES są podatne na **atak z pomiarem czasu** (ang. *timing attack*), w którym atakujący ustala informacje dotyczące szyfrowanych danych, mierząc czas szyfrowania. Atakujący musi mieć jednak dostęp do określonego komputera, na którym jest wykonywane szyfrowanie, nie jest to więc w istocie niedostatek szyfrowania, niemniej w razie naruszenia bezpieczeństwa robi się nieprzyjemnie.

Najlepiej rozeznana słabość AES jest określana jako **atak z powiązanym kluczem** (ang. *related-key attack*). Gdy dwa klucze są matematycznie powiązane w pewien sposób, atakujący może czasami wykorzystać wiedzę zgromadzoną na podstawie komunikatów szyfrowanych jednym kluczem, aby ujawnić komunikat zaszyfrowany drugim kluczem. Badacze odkryli sposób odtworzenia klucza szyfrowania AES dla pewnego kryptogramu w krótszym czasie niż atak siłowy, lecz metoda ta wymaga kryptogramów otrzymanych przez szyfrowanie tego samego tekstu jawnego kluczami powiązanymi z kluczem oryginalnym w bardzo specyficzny sposób.

Choć ten skrót kwalifikuje się jako złamanie szyfru (ang. *crack*), może nie mieć praktycznego znaczenia dla atakujących. Przede wszystkim, mimo że znacznie zmniejsza ilość pracy potrzebnej do odtworzenia pierwotnego klucza, może się nie nadawać do żadnego z istniejących komputerów lub sieci komputerowych. Po drugie, niełatwo jest uzyskać owe inne kryptogramy zaszyfrowane kluczami powiązanymi; wymaga to spenetrowania implementacji lub użycia danego szyfru. Dlatego takie złamanie szyfru jest obecnie traktowane jako teoretyczna, a nie praktyczna słabość systemu.

Być może najbardziej niepokojącym aspektem takiego złamania szyfru jest to, że uważa się, iż zadziała ono tylko dla rzekomo silniejszej, 256-bitowej wersji AES, a nie dla prostszej wersji 128-bitowej opisanej w tym rozdziale. W tym przejawia się być może największa słabość współczesnych metod kryptograficznych — ich złożoność. Wady mogą latami zostawać niewykryte mimo wysiłków doświadczonych analityków. Małe zmiany w projekcie mogą mieć wielki wpływ na bezpieczeństwo, a rozwiązania wprowadzane z zamiarem jego zwiększenia mogą wywołać odwrotne skutki.

Ograniczenia szyfrowania z kluczem symetrycznym

Rzeczywiste ograniczenie metody szyfrowania takiej jak AES nie ma natomiast nic wspólnego z potencjalnymi, ukrytymi wadami.

Wszystkie metody szyfrowania przedstawione w tym rozdziale, łącznie z AES, są określane jako metody z **kluczem symetrycznym** (ang. *symmetric key*). Oznacza to, że klucz, który szyfruje komunikat lub plik, jest taki sam jak klucz używany do jego odszyfrowania. Jeśli chcesz użyć AES do zaszyfrowania pliku na twardym dysku w swoim komputerze lub listy kontaktów w swoim telefonie, nie stwarza to problemu — tylko Ty zamykasz i otwierasz dane. Co będzie jednak, gdy potrzebujesz bezpiecznego przesyłania danych podczas wprowadzania numeru karty kredytowej w sklepie internetowym? Możesz zaszyfrować dane szyfrem AES i wysłać je na witrynę, lecz oprogramowanie witryny nie zdoła ich zdeszyfrować bez klucza.

Na tym właśnie polega **problem dzielonego** (wspólnego) **klucza** (ang. *shared key problem*) i jest on jednym z najistotniejszych problemów kryptografii. Bez bezpiecznego sposobu przekazywania kluczy szyfrowanie z kluczem symetrycznym, z samej swojej natury, nadaje się tylko do zamykania własnych, prywatnych danych. Szyfrowanie danych w celu ich przesyłania wymaga innego podejścia — użycia różnych kluczy do szyfrowania i do deszyfrowania. Jak to się robi, zobaczysz w rozdziale 3.

Wcześniej jednak musimy zająć się innym problemem. AES wymaga jako klucza olbrzymiej liczby dwójkowej, a przecież nie można oczekiwać, że użytkownicy zapamiętają łańcuch 128 bitów. Zamiast tego zapamiętujemy hasła. Jak się okazuje, z bezpiecznym przechowywaniem i użytkowaniem haseł też jest niemało kłopotu. Temu zagadnieniu poświęcimy uwagę w następnym rozdziale.

Skorowidz

A

B

C

D

E

F

G

H

I

J

K

L

Ł

M

N

O

P

Q

R

S

Ś

T

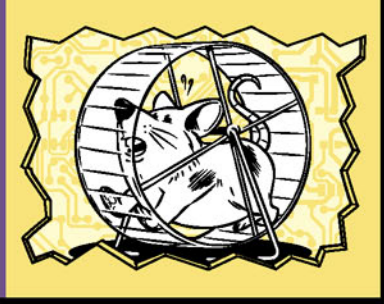
U

V

W

X

Z



SPRAWDŹ, JAK BARDZO FASCYNUJĄCE SĄ TAJNIKI OPROGRAMOWANIA!

Zawrotny rozwój technologii informatycznych sprawia, że coraz więcej osób chce poznać zasady działania oprogramowania, zwłaszcza tego najpopularniejszego. Bez znajomości pewnych zjawisk łatwo paść ofiarą tych, którzy już tę wiedzę posiedli. Żeby ją posiąść, wcale nie trzeba ukończyć studiów technicznych!

Książka, którą trzymasz w dłoni, opisuje działanie różnych rodzajów oprogramowania. Autor w przystępny i interesujący sposób wyjaśnia trudne i złożone kwestie. Nie musisz być informatykiem ani znać podstaw programowania, aby zrozumieć procesy, które przebiegają w magicznie lśniących układach scalonych, skrytych pod obudową komputera czy smartfona. Ta książka będzie Twoim przewodnikiem!

Dowiedz się, jak działa oprogramowanie:

- do szyfrowania i kryptografii z kluczem publicznym
- do obsługi haseł i podpisów cyfrowych
- do animacji i renderowania obrazu, a także do tworzenia grafiki gier
- do kompresji danych, w tym obrazów (JPEG) i wideo (MPEG-2)
- do wyszukiwania informacji w wielkich zbiorach danych
- do nawigacji i obsługi map

V. Anton Spraul — od kilkunastu lat wykłada informatykę dla studentów i początkujących programistów. Jest autorem kilku bardzo popularnych książek, znakomicie ułatwiających zrozumienie zasad programowania. Spraul niezwykle chętnie dzieli się swoją wiedzą. Prowadzi bloga, publikuje filmy na YouTubie. Mieszka w Birmingham w stanie Alabama, a jego pasje, poza informatyką, to tworzenie muzyki i gry komputerowe. Do gier używa sprzętu, który sam zbudował!

ISBN 978-83-283-2593-7
9 788328 325937
cena: 49,00 zł

no starch press