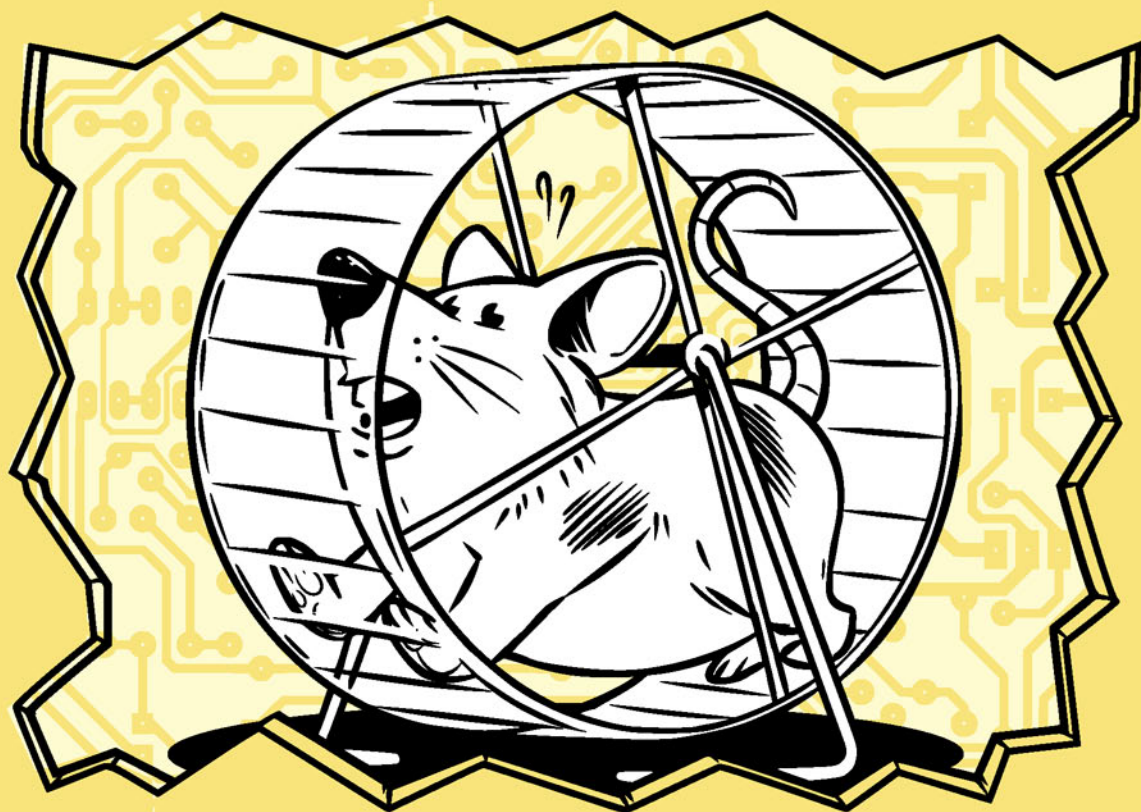


JAK DZIAŁA OPROGRAMOWANIE?

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

V. ANTON SPRAUL



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

Copyright © 2015 by V. Anton Spraul

Title of English-language original: How Software Works, ISBN 978-1-59327-666-9, published by No Starch Press.

Polish-language edition copyright © 2016 by Helion S.A.
All rights reserved.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

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)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
<http://helion.pl/user/opinie/jakdzo>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

O autorze	9
O recenzencie technicznym	9
Podziękowania	11
Wstęp	13
I	
SZYFROWANIE	17
Cel szyfrowania	18
Przestawianie — te same dane, różny porządek	19
Klucze szyfrów	20
Łamanie szyfrów	22
Podstawianie — zastępowanie danych	23
Zmienianie wzorca podstawiania	23
Poszerzanie klucza	25
Zaawansowany standard szyfrowania	26
Podstawy dwójkowe	27
Szyfrowanie AES w ujęciu ogólnym	29
Poszerzanie klucza w AES	30
Rundy szyfrowania AES	31
Łańcuchowanie bloków	33
Dlaczego AES jest bezpieczny	33
Możliwe ataki na AES	35
Ograniczenia szyfrowania z kluczem symetrycznym	36
2	
HASŁA	37
Przekształcanie hasła w liczbę	38
Cechy dobrych funkcji haszowania	38
Funkcja skrótu MD5	39
Kodowanie hasła	39
Operacje bitowe	40

Rundy haszowania MD5	42
Spełnienie kryteriów dobrej funkcji haszowania	43
Podpisy cyfrowe	43
Problem tożsamości	44
Ataki z wykorzystaniem kolizji	44
Hasła w systemach uwierzytelniania	45
Zagrożenia dotyczące tablic haseł	45
Haszowanie haseł	46
Ataki słownikowe	47
Tablice haszowania	48
Łącuchowanie haszowania	48
Haszowanie iteracyjne	51
Solenie haseł	52
Czy tablice haseł są bezpieczne	53
Usługa przechowywania haseł	54
Przemyslenia końcowe	55

3

BEZPIECZEŃSTWO W SIECI 57

Jak kryptografia z kluczem publicznym rozwiązuje problem wspólnego klucza	58
Matematyczne narzędzia kryptografii z kluczem publicznym	59
Funkcje odwracalne	59
Funkcje jednokierunkowe	60
Funkcje z bocznym wejściem	60
Metoda szyfrowania RSA	63
Tworzenie kluczy	63
Szyfrowanie danych za pomocą RSA	65
Efektywność RSA	66
Zastosowanie szyfru RSA w rzeczywistym świecie	68
Użycie RSA do uwierzytelniania	71
Bezpieczeństwo w Sieci — protokół HTTPS	73
Wymiana potwierżeń	74
Przesyłanie danych protokołem HTTPS	75
Czy problem wspólnego klucza został rozwiązany?	77

4

FILM CGI 79

Oprogramowanie tradycyjnej animacji	81
Jak działają obrazy cyfrowe	81
Sposoby definiowania kolorów	83
Jak oprogramowanie wykonuje animacje celuloidowe	84
Od oprogramowania animacji celuloidowej do renderowanej grafiki 2D	91
Oprogramowanie trójwymiarowej grafiki CGI	92
Jak opisuje się sceny trójwymiarowe	92
Kamera wirtualna	93

Oświetlenie bezpośrednie	93
Oświetlenie całego planu	98
Jak śledzić światło	99
Wygładzanie krawędzi w całej scenie	103
Łączenie rzeczywistego ze sztucznym	104
Ideal renderowania z jakością filmową	105

5

GRAFIKA GIER 107

Sprzęt do grafiki tworzonej w czasie rzeczywistym	108
Dlaczego w grach nie stosuje się śledzenia promieni	109
Same odcinki i żadnych krzywych	110
Rzutowanie bez śledzenia promieni	110
Renderowanie trójkątów	112
Algorytm malarza	113
Buforowanie głębokości	113
Oświetlanie w czasie rzeczywistym	115
Cienie	117
Światło otaczające i jego pochłanianie	118
Nanoszenie tekstur	120
Próbkowanie metodą najbliższego sąsiada	121
Filtrowanie dwuliniowe	123
Mipmapy	124
Filtrowanie trójliniowe	125
Odbicia	126
Fabrykowanie krzywizn	128
Oszukiwanie na dużych odległościach	129
Odzwzorowywanie wypukłości	129
Mozaikowanie	130
Wygładzanie krawędzi w czasie rzeczywistym	132
Superpróbki	132
Wieloprobki	134
Poprocesowe wygładzanie krawędzi	135
Budżet obrazowania	136
Co jeszcze w związku z grafiką gier	137

6

KOMPRESJA DANYCH 139

Kodowanie długości serii	141
Kompresja słownikowa	142
Podstawowa metoda	143
Kod Huffmana	144
Reorganizacja danych w celu lepszej kompresji	146
Kodowanie z przewidywaniem	146
Kwantyzacja	147

Obrazy w formacie JPEG	148
Inny sposób zapamiętywania kolorów	148
Dyskretna transformacja kosinusowa	149
DCT w dwóch wymiarach	152
Kompresowanie wyników	156
Jakość obrazów JPEG	159
Kompresja wideo wysokiej rozdzielczości	162
Nadmiarowość czasowa	162
Wideokompresja MPEG-2	163
Jakość wideo z kompresją czasową	166
Terażniejszość i przyszłość wideokompresji	168

7

WYSZUKIWANIE 169

Zdefiniowanie problemu wyszukiwania	170
Porządkowanie danych	170
Sortowanie przez wybór	170
Sortowanie szybkie	171
Wyszukiwanie binarne	175
Indeksowanie	176
Haszowanie	178
Przeszukiwanie Sieci	181
Klasyfikacja wyników	182
Efektywne zastosowanie indeksów	184
Co dalej w związku z wyszukiwaniem w Sieci	185

8

WSPÓLBIEŻNOŚĆ 187

Po co ta współbieżność?	187
Wydajność	188
Środowiska z wieloma użytkownikami	188
Wielozadaniowość	188
Kiedy współbieżność zawodzi	189
Jak działać współbieżnie i bezpiecznie	192
Dane tylko do czytania	192
Przetwarzanie transakcyjne	193
Semafor	194
Problem nieskończonego oczekiwania	196
Kolejki uporządkowane	196
Głodzenie wynikające z czekania cyklicznego	196
Kwestie sprawności semaforów	199
Co jeszcze w związku ze współbieżnością	200

9

TRASY NA MAPACH	203
Czym jest mapa w rozumieniu oprogramowania	203
Wyszukiwanie najpierw najlepszej	205
Ponowne wykorzystanie wcześniejszych wyników wyszukiwania	209
Jednoczesne znajdowanie wszystkich najlepszych tras	211
Algorytm Floyda	212
Zapamiętywanie kierunków tras	215
Przyszłość wytyczania tras	218
 SKOROWIDZ	 219

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 łózkami przechowywaliśmy pudełka po butach pełne fotografiami. 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 podarunkowe¹.

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.

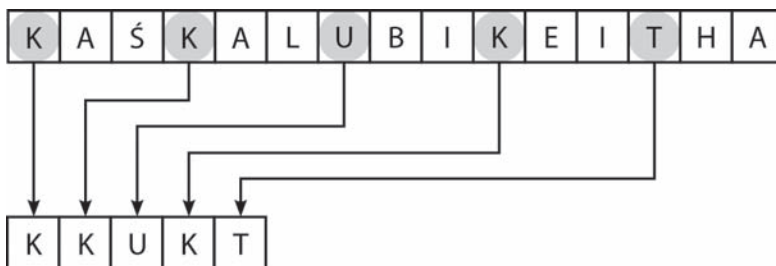
Zamiast skakać na główkę w odmęty szyfrowania opartego na oprogramowaniu, rozpocznę ten rozdział od prostych przykładów z czasów przedkomputerowych, gdy w grę wchodziły kody i zainteresowani nimi szpiegdy. Mimo że siła szyfrowania z biegiem lat niepomiaralnie wzrosła, u podstaw każdego szyfrowania leżą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

² Tu i w 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

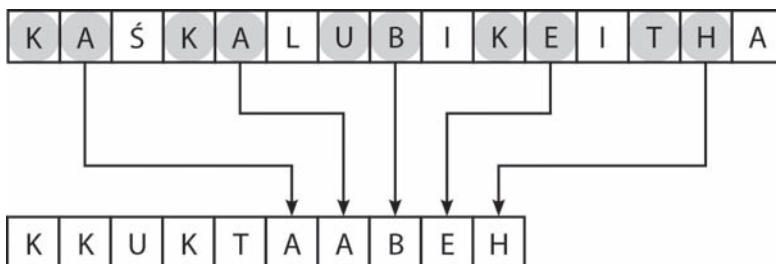
Jeden z najprostszych sposobów szyfrowania danych nazywa się **przestawianiem** (transpozycją, ang. *transposition*) i polega po prostu na „zmienianiu pozycji”. Przestawianie jest rodzajem szyfrowania, którego moi koledzy i ja używaliśmy, przekazując sobie liściki w czasach szkolnych. Ponieważ te liściki 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

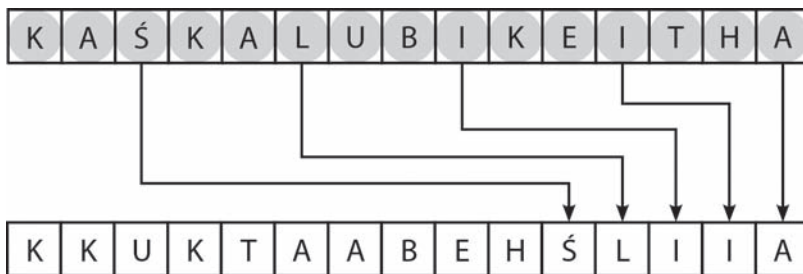
Po dotarciu do końca wiadomości wróciłem do początku i kontynuowałem wybieranie każdej co trzeciej litery z pozostałych. W drugim etapie otrzymałem stan przedstawiony na rysunku 1.2.



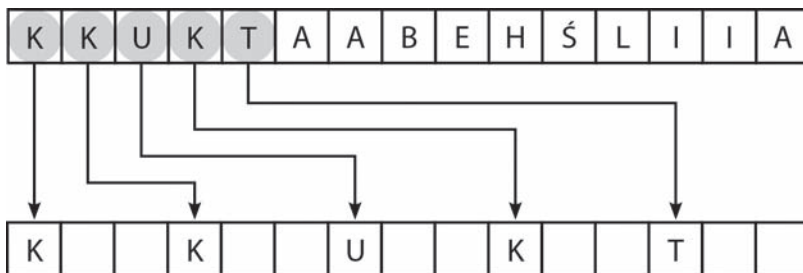
Rysunek 1.2. Drugi etap transpozycji

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.

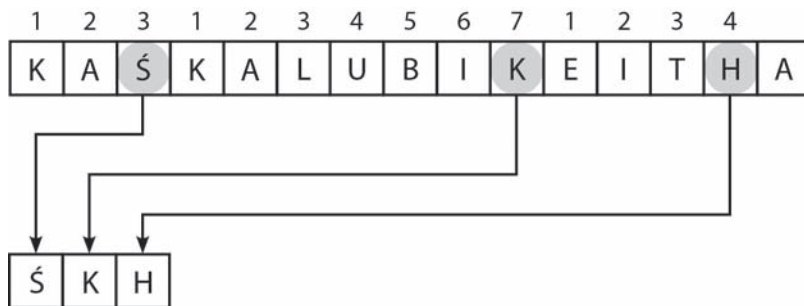
Z tego powodu wszystkie dobre systemy szyfrowania spełniają regułę określoną przez jednego z pierwszych kryptografów, Duńczyka Auguste’a Kerckhoffs’a, znaną jako **zasada Kerckhoffs’a**: bezpieczeństwo danych nie powinno zależeć od utrzymania metody szyfrowania w tajemnicy.

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.

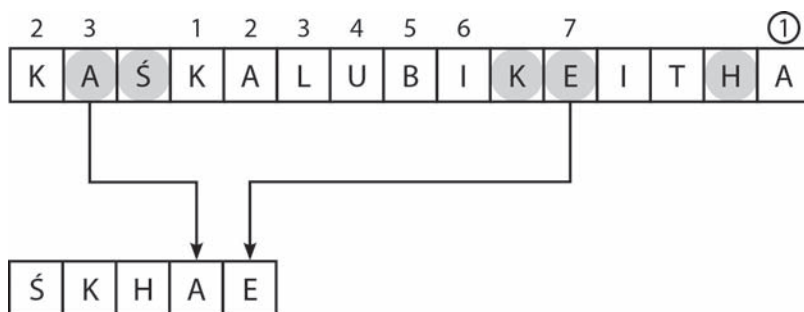
W tej metodzie nadawcy i odbiorcy, zanim zaczną przysyłać jakiegokolwiek komunikaty, wchodzi w posiadanie wspólnej tajnej liczby. Powiedzmy, że moi koledzy i ja umówiliśmy się, że będzie to 374. Liczby tej użyjemy do zmiany

wzorca przestawiania w naszych kryptogramach, Dla wiadomości KAŚKA LUBI KEITHA wzorec 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 liter³. 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 krybami⁴. 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. łódeczko, żłóbek, oszalowanie, ściaga, 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.

Zmianie wzorca podstawiania

Aby uodpornić szyfrowanie na analizę frekwencyjną, możemy zmieniać wzorec 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

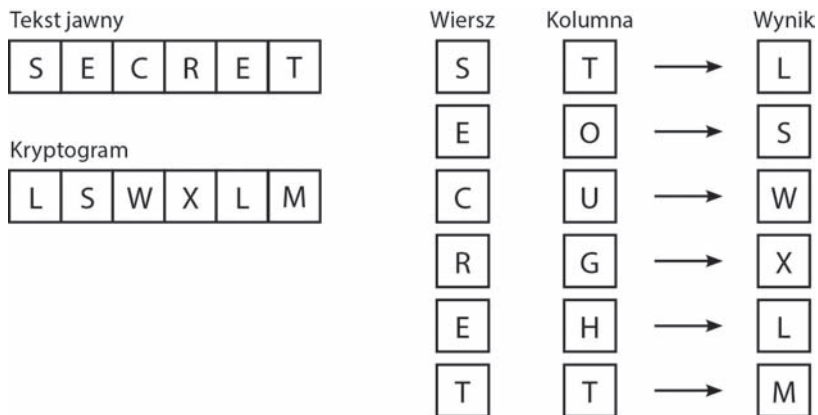
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	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 (zaciemnione pierwsza kolumna i pierwszy wiersz zawierają etykiety)

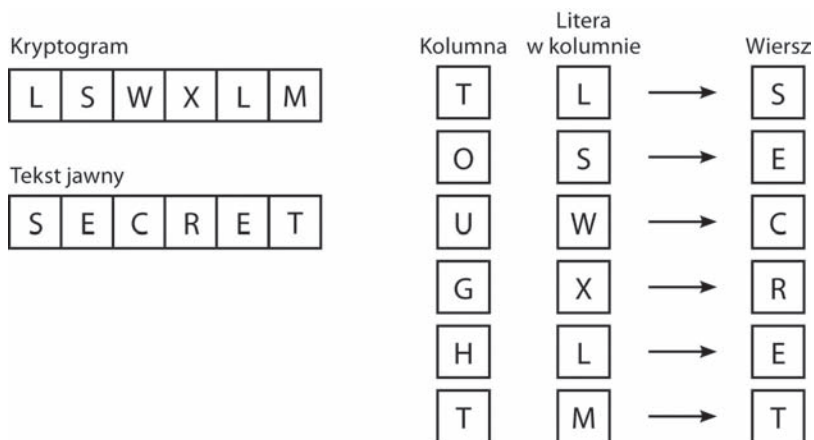
tabeli każdy wiersz i kolumna są poetykietowane literą alfabetu, która rozpoczyna dany wiersz lub kolumnę. Każde oczko w siatce jest lokalizowane za pomocą dwóch liter, na przykład w wierszu D i kolumnie H występuje litera K.

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 wzorec 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 zaszyfowanego, 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 osiągnięcie 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 wymienić 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.: „[...] kiedyś był w domu, lepiej mi było; ale podróżny musi przestać na małym” (przeł. L. Urlich. W. Szekspir, *Dzieła dramatyczne*, tom II, *Komedie*. PIW, Warszawa 1958) — *przyp. tłum.*

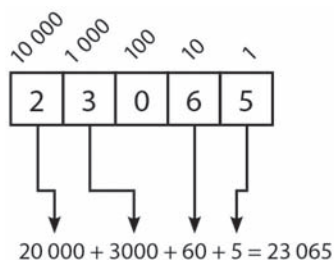
Zaawansowany standard szyfrowania (AES, ang. *Advanced Encryption Standard*) jest standardem otwartym, co oznacza, że jego specyfikacja może być urzeczywistniona przez każdego bez opłacania licencji. Czy sobie z tego zdajesz sprawę, czy nie, większość Twoich danych jest chroniona z użyciem AES. Jeśli 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.

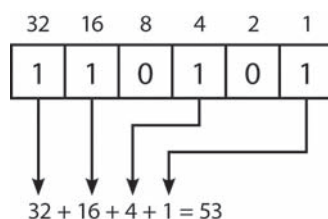
System dziesiętny a system dwójkowy

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 jednostki

W **dwójkowym** (binarnym, ang. *binary*) systemie liczbowym istnieją tylko dwie cyfry: 0 i 1, nazywane **bitami** od angielskiego *binary digits* (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 jednostki

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 dziesiątne 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 przełączają⁷ wartości odpowiednich bitów pierwszej liczby, jak pokazano na rysunku 1.12.

Wartość początkowa	1	0	0	1	1	1	0	0
XOR-owana z	0	0	1	1	0	1	1	0
daje w wyniku	1	0	1	0	1	0	1	0

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.*

Wartość początkowa	1	0	0	1	1	1	0	0
XOR-owana z	0	0	1	1	0	1	1	0
daje w wyniku	1	0	1	0	1	0	1	0
który XOR-owany z	0	0	1	1	0	1	1	0
przywraca	1	0	0	1	1	1	0	0

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 szesnastki bajtów, a cienkie linie oddzielają bity w bajtach.

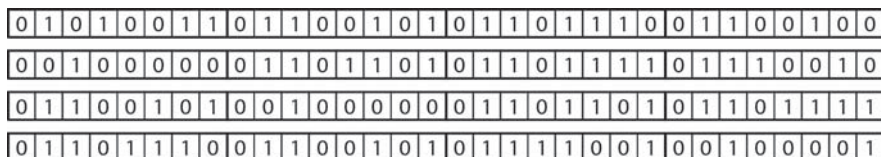
Dane tekstu jawnego są dzielone na tyle 16-bajtowych bloków, ile trzeba. Jeśli ostatni blok nie jest pełny, jego resztę uzupełnia się losowymi liczbami dwójkowymi.

⁸ Z ang. przyslijcie 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

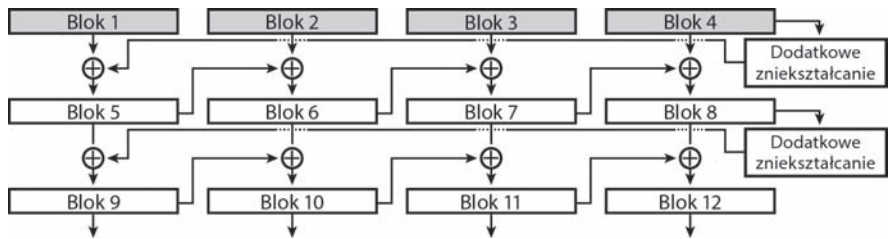
AES poddaje następnie każdy 16-bajtowy blok 10 **rundom** (ang. *rounds*) szyfrowania. W jednej rundzie bajty są przestawiane wewnątrz bloku i podstawiane z użyciem tabeli. Potem za pomocą operacji XOR bajty bloku są łączone ze sobą i z jednym ze 128-bitowych kluczy.

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 czterech



Rysunek 1.15. Proces poszerzania klucza w AES

rech bloków zacięniowanych 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 wzorec bitowy	Zastępujący wzorec 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

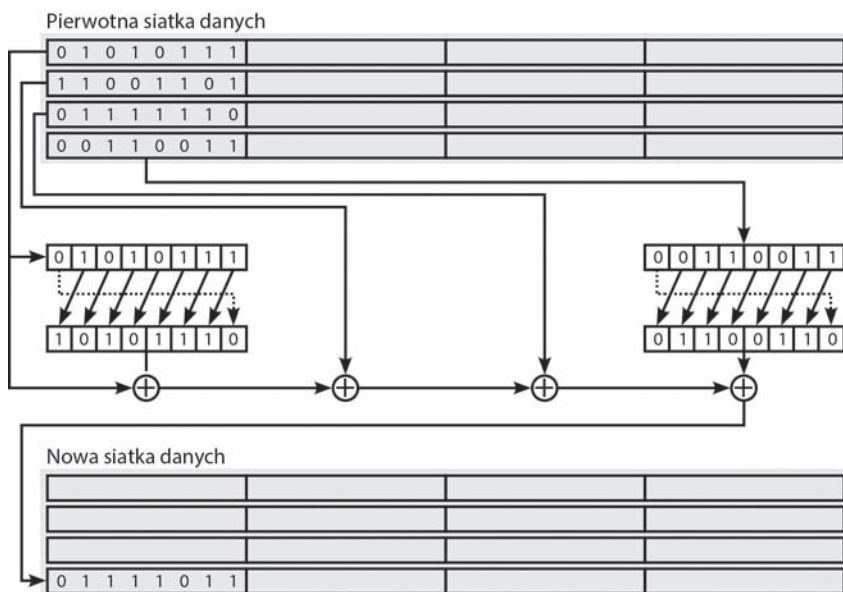
Każdy z 16 bajtów w siatce jest zastępowany z użyciem tej samej tablicy S-boks, która była użyta do poszerzania klucza.

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.

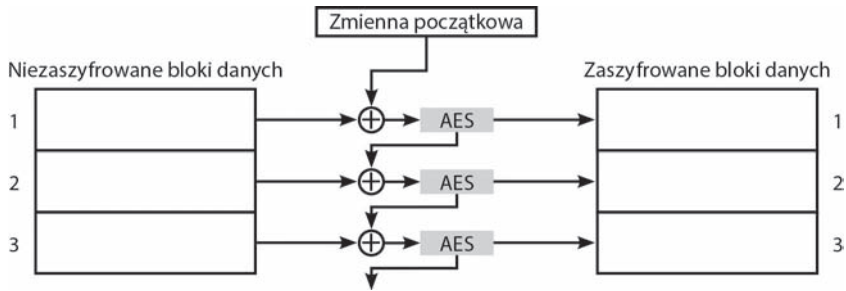
Z tego powodu systemy szyfrowania blokowego, takie jak AES, są modyfikowane, aby z identycznych bloków w tekście jawnym powstawały inne bloki w kryptogramie. Jedną z takich modyfikacji zwie się **łańcuchowaniem bloków** (ang. *block chaining*).

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 możliwych kluczy, a w ataku siłowym, aby móc oczekiwać trafienia na właściwy, należałoby uwzględnić około połowy z nich. Atakujący z komputerem mogącym wypróbować milion kluczy na sekundę zdołałby wypróbować dziennie $1\,000\,000 \times 60$ (sekund) $\times 60$ (minut) $\times 24$ (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 wysłać 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 powtórnego 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 umożliwiłyby 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 rozpoznana słabość AES jest określana jako **atak z powiązaniem 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 powiązаныmi 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 powiązаныmi; 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 twardego 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.

Weześniej 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 hasel też jest niemało kłopotu. Temu zagadnieniu poświęcimy uwagę w następnym rozdziale.

Skorowidz

A

Advanced Encryption Standard, *Patrz:* AES
AES, 27, 29, 31, 33, 69, 76
 deszyfrowanie, 33
 podatność na ataki, 35, 77
 poszerzanie klucza, 29, 30
 runda, 30
 zastosowania, 69
algorytm
 deflacji, *Patrz:* deflacja
 Floyda, 212
 Floyda-Warshalla, *Patrz:* algorytm Floyda
 malarza, 113
 wyszukiwanie najpierw najlepszej, 205, 206, 209
all-pairs shortest paths, *Patrz:* problemem
 najkrótszych ścieżek między wszystkimi parami
alpha channel, *Patrz:* kanał alfa
alpha level, *Patrz:* poziom alfa
alternatywa
 włączająca, 41, *Patrz też:* OR
 wykluczająca, 28, *Patrz też:* XOR
ambient light, *Patrz:* światło otaczające
ambient occlusion, *Patrz:* okluzja dookólna
American Standard Code for Information
 Interchange, *Patrz:* ASCII
analiza frekwencyjna, 23, 26, 33
AND, 42
animacja celuloidowa, 81, 84, 91
anti-aliasing, *Patrz:* model rastrowanie wygładzanie
 krawędzi
ASCII, 29, 38
asymmetric-key, *Patrz:* klucz asymetryczny
atak
 człowiek pośrodku, 74, 77
 kolizyjny, 44

 siłowy, 63
 słownikowy, 47
 z pomiarem czasu, 35
 z powiązaniem kluczem, 35
 z wykorzystaniem konfliktu, *Patrz:* atak
 kolizyjny
 ze znanym tekstem jawnym, 22
atakujący, 18, 22
attacker, *Patrz:* atakujący
authentication, *Patrz:* uwierzytelnienie
avalanche property, *Patrz:* efekt lawiny

B

bajt, 27
best-first search, *Patrz:* algorytm wyszukiwanie
 najpierw najlepszej
bezpieczeństwo komputerowe, 17
bezwładność widzenia, 81
B-Frame, *Patrz:* klatka dwukierunkowa
bidirectional frame, *Patrz:* klatka dwukierunkowa
bilinear filtering, *Patrz:* filtrowanie dwuliniowe
bit, 27, 28
bitmapa, 83, 84, 107, 120
 rozdzielczość, 83, 118
bitwise operations, *Patrz:* operacja bitowa
bitwise rotation, *Patrz:* rotacja bitowa
block chaining, *Patrz:* łańcuchowanie bloków
bufor
 głębokości, 113, 114, 117
 wyświetlania, 83, 117
bump mapping, *Patrz:* renderowanie
 odwzorowywanie wypukłości
byte, *Patrz:* bajt

C

cel animation, *Patrz:* animacja celuloidowa
central processing unit, *Patrz:* CPU
certyfikat, 74, 75
CGI, 79
chain merging, *Patrz:* łańcuch haszowania
zlewanie się
cieniowanie pikseli, 115
cipher key, *Patrz:* klucz
ciphertext, *Patrz:* kryptogram
clear reflection, *Patrz:* odbicie czyste
code book, *Patrz:* książka kodowa
collision, *Patrz:* funkcja haszowania kolizja
collision attack, *Patrz:* atak kolizyjny
composite number, *Patrz:* liczba złożona
concurrency, *Patrz:* współbieżność
coprime number, *Patrz:* liczba względnie pierwsza
CPU, 108
cut scene, *Patrz:* przerywnik

D

dane
analogowe, 139
cyfrowe, 140
kompresja, *Patrz:* kompresja
kwantowanie, *Patrz:* kwantyzacja
nienumeryczne, 28
obrazowe, 141
porządkowanie, 170
sortowanie, 170, 178, 180
liczba osiowa, 171
przez wybór, 170
szybkie, 171, 175
uporządkowane, 175
zbiór, *Patrz:* zbiór danych
dynamiczny, 179
statyczny, 179
DCT, 149, 150, 153
deadlock, *Patrz:* współbieżność semafor
zakleszczenie
decryption, *Patrz:* deszyfrowanie
deflate, *Patrz:* deflacja
depth buffering, *Patrz:* rzutowanie buforowanie
głębokości
deszyfrowanie, 18, 24, 33
dictionary, *Patrz:* słownik
dictionary attack, *Patrz:* atak słownikowy
dictionary compression, *Patrz:* kompresja
słownikowa

diffuse reflection, *Patrz:* odbicie rozproszone
digital composition, *Patrz:* składanie cyfrowe
digital image, *Patrz:* obraz cyfrowy
directed graph, *Patrz:* graf skierowany
discrete cosine transform, *Patrz:* DCT
display buffer, *Patrz:* bufor wyświetlania
distant impostor, *Patrz:* renderowanie oszukiwanie
w odległości
driver, *Patrz:* moduł sterujący
drukarka optyczna, 105
dyfuzja, 34

E

efekt lawiny, 34
ekran
cielkokrystaliczny, *Patrz:* ekran LCD
LCD, 82, 83
encryption, *Patrz:* szyfrowanie
environment mapping, *Patrz:* renderowanie
odzworowanie otoczenia
exclusive-or, *Patrz:* XOR

F

factor, *Patrz:* liczba złożona czynnik
fast approximate anti-aliasing, *Patrz:* FXAA
film
animowany, 81
klatka, *Patrz:* klatka
filtr rozblokowujący, 167
filtrowanie
dwuliniowe, 123, 124
trójliniowe, 125, 126
format
H.264, 168
JPEG, 148, 152, 158
jakość, 159, 160, 162
MPEG-2, 163, 166
MPEG-4, 168
TGA, 141, 142
zip, 146
frame, *Patrz:* klatka
frequency analysis, *Patrz:* analiza frekwencyjna
full-screen anti-aliasing, *Patrz:* renderer
pełnoekranowe wygładzanie krawędzi
function, *Patrz:* funkcja
funkcja
haszowania, 38, 43, 179
kolizja, 38, 44, 180, 181
MD5, *Patrz:* MD5

nieodwracalność, 38
tablica, 179, 180, 181
jednokierunkowa, 60, 61
odwracalna, 59
redukcji, 48
skrótów, *Patrz:* funkcja haszowania
z bocznym wejściem, 61, 63
FXAA, 135, 136

G

geographic information system, *Patrz:* GIS
GIS, 218
global illumination model, *Patrz:* oświetlenie
model globalny
Google Maps, 203
GOP, 163
gra online dla wielu graczy, *Patrz:* MMO
graf skierowany, 204
grafika
2D, 91
3D, 92, 107
bitmapowa, 107
dwuwymiarowa, *Patrz:* grafika 2D
trójwymiarowa, *Patrz:* grafika 3D
group of pictures, *Patrz:* GOP

H

handshaking, *Patrz:* wymiana potwierżeń
hash chain, *Patrz:* łańcuch haszowania
hash chaining, *Patrz:* haszowanie łańcuchowanie
hash code, *Patrz:* kod haszowania
hash function, *Patrz:* funkcja haszowania
hash value, *Patrz:* wartość haszowania
hashing, *Patrz:* haszowanie
hasło, 37, 45
haszowanie, 38
solenie, 52, 53, 54
usługa przechowywania, 54
hasz, 38
haszowanie, 38, 179
iteracyjne, 51, 53, 54
łańcuchowanie, 48, 51
zlewanie się, 51
runda, 42
tablica, 179, 180, 181
zatraskowe, 77
height map, *Patrz:* mapa wysokości
Huffmana kodowanie, *Patrz:* kod Huffmana

I

ID, 73
IDCT, 152
I-Frame, *Patrz:* klatka wewnętrznie kodowana
I-klatka, *Patrz:* klatka wewnętrznie kodowana
inbound link, *Patrz:* wyszukiwarka sieciowa
doprowadzenie
inclusive-OR, *Patrz:* OR
indeksowanie, 177, 178
obrazów, 185
intracoded frame, *Patrz:* klatka wewnętrznie
kodowana
inverse discrete cosine transform, *Patrz:* IDCT
invertible function, *Patrz:* funkcja odwracalna
issuer, *Patrz:* certyfikat wystawca
iterative hashing, *Patrz:* haszowanie iteracyjne

J

jednostka
centralna, *Patrz:* CPU
cieniowania pikseli, 115, 116, 120

K

kamera, 93
obiektów, *Patrz:* obiektów
kanał alfa, 90
kąć
padania światła, 96
reflektancji, *Patrz:* kąć widzenia
widzenia, 96
Kerckhoffs Auguste, 20
Kerckhoffsza zasada, *Patrz:* zasada Kerckhoffsza
key expansion, *Patrz:* klucz poszerzanie
key size, *Patrz:* klucz rozmiar
keyed hashing, *Patrz:* haszowanie zatraskowe
keyframe, *Patrz:* klatka kluczowa
klatka, 81
dwukierunkowa, 163, 164
kluczowa, 81
pośrednia, 81
generowanie, *Patrz:* twining
przewidywana, 163, 164
wewnętrznie kodowana, 163, 164
klient, 73
klucz, 20
AES, 29, 30
asymetryczny, 58
dzielony, 36, 57, 58

- klucz
 - jawny, *Patrz:* klucz publiczny
 - MAC, 77
 - poszerzanie, 26
 - prywatny, 58, 71
 - publiczny, 58, 63, 64, 71, 77
 - rekordu, *Patrz:* rekord klucz
 - rozmiar, 38
 - symetryczny, 36, 71
 - wspólny, *Patrz:* klucz dzielony
 - known-plaintext attack, *Patrz:* atak ze znanym tekstem jawnym
 - kod
 - ASCII, *Patrz:* ASCII
 - haszowania, 38, 44, *Patrz też:* hasz
 - Huffmana, 144, 146, 147, 156, 158, 159
 - predykcyjny, 147
 - prognozujący, *Patrz:* kod predykcyjny
 - przedrostkowy, 145
 - uwierzytelniający komunikatu, *Patrz:* MAC
 - kodowanie długości serii, 141
 - kolor
 - luminancja, 148, 159
 - mieszanie
 - addytywne, 82, 99
 - subtraktywne, 82, 99
 - różnica
 - czerwona, 148
 - niebieska, 148
 - system RGB, *Patrz:* RGB
 - kompresja
 - bezstratna, 140, 141
 - czasowa, 162, 164
 - jakość, 166, 167
 - MPEG-2, 163
 - deflacja, 146
 - H.264, 168
 - JPEG, *Patrz:* format JPEG
 - MPEG-2, 166
 - MPEG-4, 168
 - słownikowa, 142, 143, 146
 - okno przesuwne, 146
 - stratna, 141, 148
 - wideo, 148, 162
 - nadmiarowość czasowa, 162, 164, 166, 167
 - komunikat
 - kod uwierzytelniający, *Patrz:* MAC
 - uwierzytelniony, 71
 - zabezpieczony, 71
 - koniunkcja, 42, *Patrz też:* AND
 - kosztu minimalizacja, 203, 205
 - kryb, 22, 33, 34, 76
 - kryptoanalityczna ściągą, *Patrz:* kryb
 - kryptografia, 35
 - z kluczem publicznym, 57, 58, 59, 71, *Patrz też:* RSA
 - kryptogram, 18
 - książka kodowa, 26
 - kwantyzacja, 147, 156
- ## L
- lawinowość, 39
 - liczba
 - dwójkowa, 27, 28, *Patrz też:* system dwójkowy
 - pierwsza, 61
 - względnie pierwsza, 61
 - złożona, 61
 - czynnik, 61
 - lighting model, *Patrz:* oświetlenie model
 - linia
 - krzywa, 84
 - prosta, 84
 - local coordinate, *Patrz:* współrzędna lokalna
 - lossless compression, *Patrz:* kompresja bezstratna
 - lossy compression, *Patrz:* kompresja stratna
- ## Ł
- łańcuch haszowania, 48, *Patrz też:* haszowanie
 - łańcuchowanie
 - zlewianie się, 51
 - łańcuchowanie bloków, 33, 34, 76
- ## M
- MAC, 76
 - MAC key, *Patrz:* klucz MAC
 - makroblok, 164
 - man-in-the-middle, *Patrz:* atak człowiek pośrodku
 - mapa, 203, 218
 - bitowa, *Patrz:* bitmapa
 - cieni, 117, 118
 - witryny, *Patrz:* witryna mapa
 - wysokości, 129, 131
 - massively multiplayer online game, *Patrz:* MMO
 - master secret, *Patrz:* tajemnica główna
 - MD5, 39, 40, 42, 43, 77
 - podatność na ataki, 44, 45
 - message authentication code, *Patrz:* MAC

metoda
 najbliższego sąsiada, 122
 odwzorowywania wypukłości, *Patrz:*
 renderowanie odwzorowywanie wypukłości
 podstawieniowa, 23
 siłowa, 33, 34
 metoda siłowa, 22
 mipmapa, 125
 MMO, 191
 model, 84
 3D, 92
 wyglądanie, 128, 129, 130, 132, 133, 134,
 135, 136
 animowanie, 85
 interpolacja, 85, 86
 rastrowanie, 87, 88
 poziom alfa, 89
 schodkowanie, 88
 wyglądanie krawędzi, 89, 103
 skalowanie, 87
 translacja, 87
 model oświetlenia, *Patrz:* oświetlenie model
 moduł sterujący, 188
 mozaikowanie, 130, 131
 MSA, 134
 multisample anti-aliasing, *Patrz:* MSA
 multitasking, *Patrz:* wielozadaniowość

N

nagrobek, 181
 nearest-neighbor sampling, *Patrz:* próbkowanie
 metodą najbliższego sąsiada
 negacja, 41
 nieprzezroczystość, 89
 normalna, 115
 odchylenie, 116, 130
 NOT, 41

O

obiektyw, 93
 obliczanie osłonięcia obszaru ekranu, *Patrz:* SSAO
 obraz
 cyfrowy, 81
 generowany komputerowo, 79
 grupa, *Patrz:* GOP
 przenikanie, 105
 strumieniowanie, 140
 odbicie
 czyste, 126
 rozproszone, 95, 96

zwierciadlane, 95, 97, 100, 102
 okluzja dookólna, 119
 one-time pad, *Patrz:* podkładka jednorazowa
 one-way function, *Patrz:* funkcja jednokierunkowa
 operacja bitowa, 28
 optical printer, *Patrz:* drukarka optyczna
 OR, 41
 osłonięcie otoczenia, *Patrz:* okluzja dookólna
 oświetlenie, 93
 bezpośrednie, 98
 lustrzane, 130
 model, 94
 globalny, 99
 odległość, 95, 96
 otaczające, 119
 pośrednie, 99, 118
 rozproszone, 96, 130

P

painter's algorithm, *Patrz:* algorytm malarza
 pamięć
 adres, 177
 bufor dzielony, 189
 dzielona, 189
 przydział
 stały, 176, 177
 zmienny, 176, 177
 persistence of vision, *Patrz:* bezwładność widzenia
 piksel, 82
 cieniowanie, *Patrz:* cieniowanie pikseli
 surowy, 141
 tekstury, *Patrz:* tekssel
 pivot, *Patrz:* dane sortowanie liczbami osiami
 pixel shader, *Patrz:* jednostka cieniowania pikseli
 pixel shading, *Patrz:* cieniowanie pikseli
 P-klatka, *Patrz:* klatka przewidywana
 plaintext, *Patrz:* tekst jawny
 plik, 28
 format, *Patrz:* format
 podpis, *Patrz:* podpis cyfrowy
 szyfrowanie, *Patrz:* szyfrowanie
 podkładka jednorazowa, 26
 podpiksel, 133
 podpis cyfrowy, 43, 75
 wiarygodność, 44
 podstawianie, *Patrz też:* metoda podstawieniowa
 proste, 23, 30
 wieloalfabetowe, 23, 24, 25
 pole widzenia, 111

polyalphabetic substitution, *Patrz:* podstawianie wieloalfabetowe

post-process anti-aliasing, *Patrz:* wygładzanie krawędzi poprocesowe

poziom alfa, 89

praca w tle, 189

precomputed hash table, *Patrz:* tablica haszowania z góry obliczona

predicted frame, *Patrz:* klatka przewidywana

predictive encoding, *Patrz:* kod predykcyjny

prefix code, *Patrz:* kod przedrostkowy

premaster secret, *Patrz:* tajemnica poboczna

prime number, *Patrz:* liczba pierwsza

private-key, *Patrz:* klucz prywatny

problem

- dzielonego klucza, 36
- wspólnego klucza, 77

procesor, 108, 188

- graficzny, 109
- operacja test-and-set, 195
- wielordzeniowy, 200

program praca w tle, 189

projection, *Patrz:* rzutowanie

protokół

- HTTPS, 73
- HTTP, 73
- HTTPS, 74, 77

próbkiwanie, 120

filtrowanie

- dwuliniowe, 123, 124
- trójliniowe, 125, 126
- metodą najbliższego sąsiada, 122

przenikanie, *Patrz:* dyfuzja

przerywnik, 108

przetwarzanie

- transakcyjne, 193

przezroczystość, 89

public-key, *Patrz:* klucz publiczny

public-key cryptography, *Patrz:* kryptografia z kluczem publicznym

punkt obrony pojedynczy, 45

rekord, 170, 176

- adres, 177
- długość, 176, 177
- klucz, 170
- usuwanie, 181

related-key attack, *Patrz:* atak z powiązonym kluczem

renderer, 91, 95, 99

- pełnoekranowe wygładzanie krawędzi, 103
- śledzenie promieni, *Patrz:* śledzenie promieni

renderowanie, 91

- bez śledzenia promieni, 105, 110
- cieniowanie, 115, 117
- mozaikowanie, *Patrz:* mozaikowanie o jakości filmowej, 92, 93, 94, 105
- odwzorowanie
 - otoczenia, 126, 127
 - wypukłości, 129, 130, 131
- oszukiwanie w odległości, 129, 130, 131
- trójkąta, 112
- w czasie rzeczywistym, 92, 103, 107, 108, 109, 115
 - bez śledzenia promieni, 110, 112
 - sprzęt, 108

RGB, 83, 140

robot, 182

rotacja bitowa, 32

rozdzielczość, 83, 118

- ultrawysoka, *Patrz:* UHD

RSA, 63, 66, 67, 68

- podatność na ataki, 77
- uwierzytelnienie, *Patrz:* uwierzytelnienie RSA zastosowania, 68, 69

run, *Patrz:* seria

run-length encoding, *Patrz:* kodowanie długości serii

rzutowanie, 93, 111

- buforowanie głębokości, 113, 114
- w śledzeniu promieni, *Patrz:* śledzenie promieni rzutowanie

Q

quantization, *Patrz:* kwantyzacja

quicksort, *Patrz:* dane sortowanie szybkie

R

race condition, *Patrz:* współbieżność wyścig

ray tracing, *Patrz:* śledzenie promieni

reduction function, *Patrz:* funkcja redukcji

S

sampling, *Patrz:* próbkiwanie

S-box, *Patrz:* tablica S-boks

scena, 93

screen coordinate, *Patrz:* współrzędna ekranowa

screen space ambient occlusion, *Patrz:* SSAO

search, *Patrz:* wyszukiwanie

selection sort, *Patrz:* dane sortowanie przez wybór

sequential search, *Patrz:* wyszukiwanie liniowe

seria, 141
serwer, 73
 uwierzytelnienie, *Patrz:* uwierzytelnienie serwera
sesja, 74, 75
session, *Patrz:* sesja
shadow map, *Patrz:* mapa cieni
shared buffer, *Patrz:* pamięć bufor dzielony
shared key, *Patrz:* klucz dzielony
shared key problem, *Patrz:* problem dzielonego klucza
Sieć
 głęboka, 182
 wyszukiwarka, *Patrz:* wyszukiwarki sieciowa
signature, *Patrz:* podpis cyfrowy
single point of defence, *Patrz:* punkt obrony pojedynczy
sitemap, *Patrz:* wityrna mapa
składanie cyfrowe, 104
sliding window, *Patrz:* kompresja słownikowa okno przesuwne
słownik, 47, 170
słowo kluczowe, *Patrz:* wityrna słowo kluczowe
Sony PlayStation, 108
sorting, *Patrz:* dane sortowanie
specular reflection, *Patrz:* odbicie zwierciadlane
Spielberg Steven, 79
spin lock, *Patrz:* wirująca blokada
spooling drukowania, 189
SSAA, 132, 133
SSAO, 119
starting variable, *Patrz:* zmienna początkowa
starvation, *Patrz:* współbieżność semafor głodzenie
subpixel, *Patrz:* podpiksel
substitution method, *Patrz:* metoda podstawieniowa
superpróbkiwanie, *Patrz:* SSAA
supersampling anti-aliasing, *Patrz:* SSAA
symmetric key, *Patrz:* klucz symetryczny
system
 dwójkowy, 27
 dziesiętny, 27
 informacji geograficznej, *Patrz:* GIS
 liczbowy, 27
 uwierzytelniania, *Patrz:* uwierzytelnienie
szyfr
 klucz, *Patrz:* klucz
 łamanie, 18, 22, 35, *Patrz też:* atak

 analiza frekwencyjna, *Patrz:* analiza frekwencyjna
 metoda siłowa, *Patrz:* metoda siłowa podstawieniowy, 23
 wskaźnik odporności, 22
szyfrowanie, 18
 AES, *Patrz:* AES
 cyfrowe, 29, 30
 dyfuzja, *Patrz:* dyfuzja
 odwracanie, *Patrz:* deszyfrowanie
 przestawianie, 19
 RSA, *Patrz:* RSA
 standard zaawansowany, *Patrz:* AES z kluczem publicznym, 77, *Patrz też:* klucz publiczny
 z kluczem symetrycznym, 36

Ś

śledzenie promieni, 100, 101, 109
 cienie, 102, 117, *Patrz też:* mapa cieni
deformacje, 101
ostrość, 101
pomijanie, 105
rzutowanie, 110, 112
 symulowanie obiektów, 101
światło, *Patrz:* oświetlenie

T

tablica
 haszowania z góry obliczona, 48
 S-boks, 31, 34
tabula recta, 23
tajemnica
 główna, 75, 76
 poboczna, 75
teksele, 120
tekst jawny, 18, 23
tekstura, 120
 mipmapa, *Patrz:* mipmapa
 piksel, *Patrz:* teksele
teksturowanie, 120
temporal compression, *Patrz:* kompresja czasowa
temporal redundancy, *Patrz:* kompresja wideo nadmiarowość czasowa
tessellation, *Patrz:* mozaikowanie
texture, *Patrz:* tekstura
texture mapping, *Patrz:* teksturowanie
timing attack, *Patrz:* atak z pomiarem czasu
tocjent, 63, 64

tombstone, *Patrz:* nagrobek
Toon Boom, 91
Toonz, 91
totient, *Patrz:* tocejnt
tożsamości weryfikowanie, 73, 74, 75
transformacja kosinusowa dyskretna, *Patrz:*
DCT
odwrotna, *Patrz:* IDCT
transpozycja, 19, 30
trapdoor function, *Patrz:* funkcja z bocznym
wejściem
twining, 81
automatyczny, 85, 86

U

UHD, 168
układ współrzędnych, 83
ultra high definition, *Patrz:* UHD
usługa przechowywania hasel, *Patrz:* hasło usługa
przechowywania
uwierzytelnienie, 37, 45, 47, 71
identyfikator, *Patrz:* ID
RSA, 71
serwera, 74

V

VR, 137

W

wartość haszowania, 38
Web search engine, *Patrz:* wyszukiwarka sieciowa
wektor, 150
weryfikowanie tożsamości, *Patrz:* tożsamość
weryfikowanie
wideoedytor, 141
wideokompresja, *Patrz:* kompresja wideo
wielopróbkowanie, *Patrz:* MSAA
wielozadaniowość, 188, 189, 192, 200
wirtualna rzeczywistość, *Patrz:* VR
wirująca blokada, 196
witryna
mapa, 182
słowo kluczowe, 182, 184
synonim, 184
wyszukiwarka, *Patrz:* wyszukiwarka witryny
world coordinate, *Patrz:* współrzędna świata

współbieżność, 187, 188, 189, 200
semafor, 194
czekanie cykliczne, 198
działanie, 195, 196, 199
głodzenie, 196, 200
zakleszczenie, 198
ziarnistość, 200
wyścig, 192
zapobieganie, 192, 193, 194, 196
współrzędna
ekranowa, 83, 111
lokalna, 84, 92
świata, 93, 111
układ, *Patrz:* układ współrzędnych
wyglądanie krawędzi, 89
budżet, 136
pełnoekranowe, 103
poprocesowe, 135
superpróbkiowaniem, *Patrz:* SSAA
szybkie przybliżanie, *Patrz:* FXAA
wielopróbkiowaniem, *Patrz:* MSAA
wymiana potwierdzeń, 74
wyszukiwanie, 169, 178
binarne, 175, 176, 177
liniowe, 170, 176, 177
wyszukiwarka
sieciowa, 169, 181, 182
doprowadzenie, 183
odsylacz pośredni, 184
rozpoznawanie obrazów, 185
słowo kluczowe, *Patrz:* witryna słowo
kluczowe
witryny, 182

X

XOR, 28, 30

Z

zakupy w sieci, 73
zasada Kerckhoffs'a, 20, 45, 51
zbiór danych, 170
zmienna początkowa, 33
znajdowanie najkrótszej ścieżki, 203
znaków kodowanie, 29

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



- 1. ZAREJESTRUJ SIĘ**
- 2. PREZENTUJ KSIĄŻKI**
- 3. ZBIERAJ PROWIZJĘ**

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>



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 posiadli. Żeby ją osiąść, 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 YouTube. Mieszka w Birmingham w stanie Alabama, a jego pasje, poza informatyką, to tworzenie muzyki i gry komputerowe. Do gier używa sprzętu, który sam zbudował!

sięgnij po WIĘCEJ



KOD KORZYŚCI

ISBN 978-83-283-2593-7



9 788328 325937

cena: 49,00 zł

Helion

księgarnia internetowa



<http://helion.pl>

zamówienia telefoniczne



0 801 339900



0 601 339900

Informatyka w najlepszym wydaniu

Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel.: 32 230 98 63
e-mail: helion@helion.pl
<http://helion.pl>

Sprawdź najnowsze promocje:
• <http://helion.pl/promocje>
Książki najchętniej czytane:
• <http://helion.pl/bestsellery>
Zamów informacje o nowościach:
• <http://helion.pl/nawosc>

