Kryptografia to dziedzina informatyki zajmująca się zagadnieniami bezpieczeństwa informacji: w szczególności kodowaniem i dekodowaniem. Zupełnie zmieniła swoje oblicze w XX wieku dzięki zastosowaniu algorytmów opartych na teorii liczb.

I. Model matematyczny kodowania i dekodowania

Rozważamy zagadnienie kryptograficzne polegające na przesłaniu wiadomości od nadawcy do obiorcy w taki sposób, by żadna trzecia (przypadkowa) osoba nie była w stanie jej odczytać.

Tekstem jawnym będziemy nazywać ciąg symboli (znaków) w pewnym języku (np. w polskim), który jest dany w procesie kodowania lub jest rezultatem w procesie dekodowania. *Wiadomość zakodowana*, czyli *szyfrogram*, to także ciąg symboli, którego elementami są znaki z tego samego alfabetu co elementy tekstu jawnego (przynajmniej w ramach naszego kursu). Jest rezultatem kodowania tekstu jawnego lub też ciągiem danym w procesie dekodowania. Przed zakodowaniem nadawca dzieli tekst jawny na tzw. jednostki. Jednostką tekstu jawnego może być pojedynczy symbol, dwójka symboli, itp. Dopiero jednostkę tekstu jawnego poddaje się procesowi kodowania otrzymując jednostkę szyfrogramu. Kolejne jednostki szyfrogramu mogą być niezależnie wysyłane od nadawcy do odbiorcy. Odbiorca poddaje otrzymany szyfrogram procesowi dekodowania dostając jednostkę tekstu jawnego.

Definicja 1. *Niech J będzie zbiorem wszystkich możliwych jednostek tekstu jawnego i szyfrogramu. Wtedy* funkcją kodującą *nazywamy $f : J \to J$ - permutację tego zbioru.*

Definicja 2. *Dla funkcji kodującej f, funkcję do niej odwrotną $f^{-1} : J \to J$ nazywamy* funkcją dekodującą. *Jest ona również permutacją.*

Definicja 3. Kryptosystemem *nazywamy parę (J, f).*

Tabelka zmiany liter na liczby

$*$	$+$	$-$	A	B	C	D	E	F	G	H	I	J	K	L	M	N
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
O	P	Q	R	S	T	U	W	V	X	Y	Z		.	?	!	
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	

Niebezpieczeństwa jednostek złożonych z jednego symbolu (analiza częstości).

II. Przykład: szyfr Cezara

Zajmiemy się (z przyczyn czysto dydaktycznych - by lepiej zrozumieć zagadnienie) przykładem bardzo prostego systemu kryptograficznego, którego, zdaniem wielu historyków, Juliusz Cezar używał do porozumiewania się ze swoimi podwładnymi.

Funkcja kodująca jest tu oparta na dodawaniu modulo rozmiar $|J|$ zbioru jednostek wiadomości:

$$f(P) = (P + b) \mod |J|,$$

dla pewnego b - parametru ze zbioru liczb naturalnych mniejszych od $|J|$. Jest to tajny klucz szyfru Cezara, który jest też kluczem funkcji dekodującej:

$$f^{-1}(C) = (C - b) \mod |J|.$$

Zatem znając b, odbiorca z łatwością dekoduje szyfrogram.

Przykład *Gallia est omnis divisa in partes tres* ($b = 23$).

Pierwsza wada szyfru Cezara: łatwo rozkodować - dla przykładowej tabeli wystarczy zbadać 33 możliwości (choć jedna jest nonsensowna), nawet bez używania analizy częstości. Dlatego nawet kodowanie par lub trójek symboli metodą Cezara nie jest odporne na szybkie złamanie.

Druga wada szyfru Cezara: jeśli Cezar używał tego samego kryptosystemu do kontaktu ze wszystkimi swoimi generałami, wtedy mogło dochodzić do sytuacji niepożądanych

np. gdy generał A wysłał Cezarowi zaszyfrowaną wiadomość, generał B przechwytując kuriera po drodze mógł ją odczytać.

III. Kodowanie RSA

Definicja 4. Klucz kodowania - K_K *to parametr(y) funkcji kodującej. Znając je można wysyłać zakodowane wiadomości.* Klucz dekodowania - K_D *to parametr(y) funkcji dekodującej. Znając je można dekodować wiadomości.*

Do niedawna uważano, że w dowolnym kryptosystemie znajomość klucza kodujacego K_K pozwala na szybkie poznanie klucza dekodującego K_D, czyli że każdy kto umie kodować wiadomości może je też dekodować (tj., że druga wada szyfru Cezara jest nieunikniona). By to zmienić, spróbujmy skonstruować następujący obiekt:

Definicja 5. Kryptosystem z kluczem publicznym *to system* (J, f, K_K, K_D)*, w którym*
a) K_K *jest powszechnie dostępny;* K_D *jest zachowywany w tajemnicy,*
b) znajomość K_K *pozwala na szybkie kodowanie jednostki tekstu, czyli obliczanie* $f(P)$ *dla* $P \in J$*,*
c) znajomość K_K *nie pozwala (bez znajomości* K_D*) na szybkie dekodowanie jednostki szyfrogramu, czyli obliczanie* $f^{-1}(C)$ *dla* $C \in J$*,*
d) znajomość K_D *daje możliwość szybkiego zdekodowania jednostki szyfrogramu.*

Szukamy zatem funkcji f, którą liczy się szybko, ale wyliczenie funkcji f^{-1} jest dużo bardziej skomplikowane. Po poprzednim rozdziale wiemy, że przykładem takiej procedury jest problem znajdowania rozkładu liczby na czynniki pierwsze. W szczególności, wymnożenie dwóch, nawet dużych, liczb pierwszych jest proste i szybkie, ale znalezienie rozkładu tego iloczynu z powrotem na czynniki składowe, bez dodatkowych informacji, jest ekstremalnie trudne. Na tym właśnie opiera się kodowanie RSA (od nazwisk pomysłodawców: Rivesta, Shamira i Adlemana).

Algorytm 1 (Kodowanie RSA)**.** *I. Wybierz (zwykle robi się to pseudolosowo) dwie bardzo duże różne liczby pierwsze (p i q). Im większe, tym trudniej złamać kod, ale też dłużej trwa proces kodowania i dekodowania.*
II. Niech $n = pq$*. Wtedy* $\varphi(n) = \varphi(p) \cdot \varphi(q) = (p-1)(q-1)$*. W podobny sposób (czyli znów pseudolosowo) wybieramy liczbę e względnie pierwszą z* $\varphi(n)$*. Znajdujemy d takie, że* $e \cdot d \equiv 1 (mod \varphi(n))$*. (np. za pomocą rozszerzonego algorytmu Euklidesa).*
III. Klucz publiczny $K_K = (n, e)$ *można spokojnie opublikować. Do własnego użytku zachowujemy* $K_D = (n, d)$*.*
IV. Zbiór wszystkich jednostek wiadomości definiujemy jako $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$*. Funkcją kodującą będzie* $f(P) = P^e mod n$*.*
Funkcją dekodującą będzie $f^{-1}(C) = C^d mod n$ *(okazuje się, że są to funkcje odwrotne - dowód w slajdach)*

Uwaga O ile rozkład dużej liczby na czynniki pierwsze jest nierealny do wykonania w rozsądnym czasie, to istnieją testy sprawdzające, czy dana liczba n jest pierwsza w czasie $O(\log^3 n)$.
Uwaga Wydaje się, że potęgowanie z resztą jest tym fragmentem algorytmu RSA, który może zająć bardzo dużo czasu. Jednakże, komputer potrafi je wykonać dość szybko, dzięki metodzie szybkiego potęgowania (tzw. potęgowania przez kwadraty) w wypadku kodowania oraz dzięki kreatywnemu zastosowaniu chińskiego twierdzenia o resztach w wypadku dekodowania.
Przykład Kodowanie i rozkodowanie Alea iacta est! dla klucza publicznego $(33, 7)$.
Uwaga W praktyce algorytm RSA jest tylko podstawą systemu szyfrowania: na przykład elementy, którym przypisano liczby 0 i 1, nigdy nie zmienią swojej postaci podczas kodowania tym algorytmem, gdyż nie zmieniają swojej wartości podniesione do dowolnej potęgi (dlatego w tabelce są im przypisane nieistotne symbole). Między innymi dlatego wiadomość przed zakodowaniem powinna zostać poddana wstępnej obróbce (tzw. *padding*): dobranie jednostkom szyfrowania odpowiednich liczb, dodanie pewnego „szumu

informacyjnego", dodanie bezsensownych sekwencji znaków na początku i końcu (by nie wskazywać położenia charakterystycznych zwrotów typu: Raport, albo Szanowny Panie, czy też podpis nadawcy). Te techniki nie wchodzą w zakres materiału wykładu.

Uwaga System RSA pozostaje bezpieczny, o ile klucze są odpowiednio długie - ich długość musi się stawać coraz większa. Obecnie używa się kluczy 1024- lub 2048-bitowych. Te ostatnie, wg. ekspertów nie powinny być możliwe do złamania w rozsądnej przyszłości (chyba, że w wyniku użycia zupełnie nowych narzędzi np. komputerów kwantowych). Największy złamany klucz (tj. klucz dla którego przedstawiono efektywny algorytm znajdowania rozkładu) w dniu 1 X 2015 miał rozmiar 768 bitów. Klucze o rozmiarze 300 bitów lub mniejszym da się złamać przy pomocy zwyczajnych komputerów i darmowego oprogramowania w kilka godzin.

Uwaga Dla optymalnego bezpieczeństwa, liczby p i q nie powinny być zbyt bliskie, ale powinny być podobnego rozmiaru w zapisie bitowym.