Jak pisać, by nie każdy mógł przeczytać

Grzegorz Kosiorowski

Uniwersytet Ekonomiczny w Krakowie

Kryptologia

Kryptologia:

Kryptologia

Kryptologia:

- Kryptografia - nauka o szyfrowaniu (kodowaniu).

Kryptologia

Kryptologia:

- Kryptografia - nauka o szyfrowaniu (kodowaniu).
- Kryptoanaliza - nauka o deszyfrowaniu (dekodowaniu).

Kryptologia

Kryptologia:

- Kryptografia - nauka o szyfrowaniu (kodowaniu).
- Kryptoanaliza - nauka o deszyfrowaniu (dekodowaniu).
- Główne narzędzie (współcześnie): teoria liczb.

Kryptologia

Kryptologia:

- Kryptografia - nauka o szyfrowaniu (kodowaniu).
- Kryptoanaliza - nauka o deszyfrowaniu (dekodowaniu).
- Główne narzędzie (współcześnie): teoria liczb.
- Wszystkie liczby na tym wykładzie są całkowite.

Tekst jawny i szyfrogram

Problem: jak przesłać wiadomość od nadawcy do odbiorcy w taki sposób, by żadna trzecia (przypadkowa) osoba nie była w stanie jej odczytać?

Tekst jawny i szyfrogram

Problem: jak przesłać wiadomość od nadawcy do odbiorcy w taki sposób, by żadna trzecia (przypadkowa) osoba nie była w stanie jej odczytać?

Tekst jawny

Tekst jawny: ciąg symboli (znaków) w pewnym języku (np. w polskim), który jest dany w procesie kodowania lub jest rezultatem w procesie dekodowania.

Tekst jawny i szyfrogram

Problem: jak przesłać wiadomość od nadawcy do odbiorcy w taki sposób, by żadna trzecia (przypadkowa) osoba nie była w stanie jej odczytać?

Tekst jawny

Tekst jawny: ciąg symboli (znaków) w pewnym języku (np. w polskim), który jest dany w procesie kodowania lub jest rezultatem w procesie dekodowania.

Szyfrogram

Wiadomość zakodowana, czyli *szyfrogram*: także ciąg symboli, którego elementami są znaki z tego samego alfabetu co elementy tekstu jawnego. Jest rezultatem kodowania tekstu jawnego lub też ciągiem danym w procesie dekodowania.

Funkcja kodująca

Funkcja kodująca

Niech J będzie zbiorem wszystkich możliwych jednostek tekstu jawnego i szyfrogramu (czyli *jednostek szyfrowania*). Wtedy *funkcją kodującą* nazywamy $f : J \longrightarrow J$ - permutację, czyli przestawienie kolejności elementów tego zbioru.

Funkcja kodująca

Funkcja kodująca

Niech J będzie zbiorem wszystkich możliwych jednostek tekstu jawnego i szyfrogramu (czyli *jednostek szyfrowania*). Wtedy *funkcją kodującą* nazywamy $f : J \longrightarrow J$ - permutację, czyli przestawienie kolejności elementów tego zbioru.

Będzie polegało to na tym, że wszędzie gdzie jest w tekście jawnym jednostka np. 'bo' w szyfrogramie znajdzie się $f(bo)$, czyli na przykład 'xf'.

Uwagi techniczne

- Zmiana alfabetu - algorytmicznie nieistotna. Natomiast by łatwiej wykonywać przekształcenia matematyczne, litery zamienimy wpierw na liczby.

Uwagi techniczne

- Zmiana alfabetu - algorytmicznie nieistotna. Natomiast by łatwiej wykonywać przekształcenia matematyczne, litery zamienimy wpierw na liczby.
- Kodowane z osobna są jednostki szyfrowania: pojedyncze symbole (a, b, c ...), częściej dwójka (aa, ab, ac ...) lub więcej symboli (powód: trudność złamania kodu). Na tym wykładzie jednostkami będą pojedyncze symbole (dla łatwości zrozumienia).

Uwagi techniczne

- Zmiana alfabetu - algorytmicznie nieistotna. Natomiast by łatwiej wykonywać przekształcenia matematyczne, litery zamienimy wpierw na liczby.
- Kodowane z osobna są jednostki szyfrowania: pojedyncze symbole (a, b, c ...), częściej dwójka (aa, ab, ac ...) lub więcej symboli (powód: trudność złamania kodu). Na tym wykładzie jednostkami będą pojedyncze symbole (dla łatwości zrozumienia).
- Czy to musi być permutacja?

Uwagi techniczne

- Zmiana alfabetu - algorytmicznie nieistotna. Natomiast by łatwiej wykonywać przekształcenia matematyczne, litery zamienimy wpierw na liczby.
- Kodowane z osobna są jednostki szyfrowania: pojedyncze symbole (a, b, c ...), częściej dwójka (aa, ab, ac ...) lub więcej symboli (powód: trudność złamania kodu). Na tym wykładzie jednostkami będą pojedyncze symbole (dla łatwości zrozumienia).
- Czy to musi być permutacja?
- Dekodowanie to odwracanie procesu f oznaczane f^{-1}. Np. każde 'xf' w poprzednim przykładzie zmieni się w $f^{-1}(xf)$, czyli 'bo'.

Uwagi techniczne

- Zmiana alfabetu - algorytmicznie nieistotna. Natomiast by łatwiej wykonywać przekształcenia matematyczne, litery zamienimy wpierw na liczby.
- Kodowane z osobna są jednostki szyfrowania: pojedyncze symbole (a, b, c ...), częściej dwójka (aa, ab, ac ...) lub więcej symboli (powód: trudność złamania kodu). Na tym wykładzie jednostkami będą pojedyncze symbole (dla łatwości zrozumienia).
- Czy to musi być permutacja?
- Dekodowanie to odwracanie procesu f oznaczane f^{-1}. Np. każde 'xf' w poprzednim przykładzie zmieni się w $f^{-1}(xf)$, czyli 'bo'.
- Zajmujemy się kryptosystemami: szyframi opartymi na tej samej ogólnej zasadzie (jedyne opłacalne podejście przy wielu użytkownikach), z co najwyżej różnymi „kluczami".

Podstawowe definicje

Funkcja kodująca

Niech J będzie zbiorem wszystkich możliwych jednostek tekstu jawnego i szyfrogramu (czyli *jednostek szyfrowania*). Wtedy *funkcją kodującą* nazywamy $f : J \longrightarrow J$ - permutację, czyli przestawienie kolejności elementów tego zbioru.

Funkcja dekodująca

Dla funkcji kodującej f, *funkcją dekodującą* nazywamy „przestawienie z powrotem", czyli funkcję do niej odwrotną $f^{-1} : J \longrightarrow J$.

Kryptosystem

Kryptosystemem nazywamy połączenie zbioru jednostek tekstu J z funkcją kodującą f.

Tabelka do przykładów szyfrowania

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

Tabelka zastępuje 33 symbole liczbami 0 do 32 (resztami z dzielenia przez 33).

Tabelka do przykładów szyfrowania

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

Tabelka zastępuje 33 symbole liczbami 0 do 32 (resztami z dzielenia przez 33). Ponieważ alfabet łaciński ma mniej niż 33 litery (wliczając spację), uzupełniliśmy tabelkę różnymi nieistotnymi znakami. Później istotne się okaże, że pierwsze kilka symboli nie jest literami.

Szyfr Cezara

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

Prosty system kryptograficzny, którego, zdaniem wielu historyków, Juliusz Cezar używał do porozumiewania się ze swoimi podwładnymi.

Szyfr Cezara

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

Prosty system kryptograficzny, którego, zdaniem wielu historyków, Juliusz Cezar używał do porozumiewania się ze swoimi podwładnymi. Znak tekstu jawnego zmienia się w szyfrogramie w znak o numerze o b wyższym (np. $b = 3$ oznacza, że zamiast A piszemy D, a zamiast I piszemy

Szyfr Cezara

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

Prosty system kryptograficzny, którego, zdaniem wielu historyków, Juliusz Cezar używał do porozumiewania się ze swoimi podwładnymi. Znak tekstu jawnego zmienia się w szyfrogramie w znak o numerze o b wyższym (np. $b = 3$ oznacza, że zamiast A piszemy D, a zamiast I piszemy L). Jeśli numer będzie „za duży” - zaczynamy liczyć od początku (np. ! zakodujemy jako

Szyfr Cezara

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

Prosty system kryptograficzny, którego, zdaniem wielu historyków, Juliusz Cezar używał do porozumiewania się ze swoimi podwładnymi. Znak tekstu jawnego zmienia się w szyfrogramie w znak o numerze o b wyższym (np. $b = 3$ oznacza, że zamiast A piszemy D, a zamiast I piszemy L). Jeśli numer będzie „za duży” - zaczynamy liczyć od początku (np. ! zakodujemy jako -).

Klucze kodowania i dekodowania szyfru Cezara

b jest jedyną liczbą, której znajomość jest potrzebna do kodowania szyfrem Cezara. Dlatego mówimy, że jest *kluczem kodowania*.

Klucze kodowania i dekodowania szyfru Cezara

b jest jedyną liczbą, której znajomość jest potrzebna do kodowania szyfrem Cezara. Dlatego mówimy, że jest *kluczem kodowania*.

Klucze

Klucz kodowania (K_K) - zestaw liczb, których znajomość pozwala wysyłać zakodowane wiadomości. *Klucz dekodowania*(K_D) - zestaw liczb, których znajomość pozwala wysyłać zakodowane wiadomości.

Na czym polega funkcja dekodująca szyfru Cezara?

Klucze kodowania i dekodowania szyfru Cezara

b jest jedyną liczbą, której znajomość jest potrzebna do kodowania szyfrem Cezara. Dlatego mówimy, że jest *kluczem kodowania*.

Klucze

Klucz kodowania (K_K) - zestaw liczb, których znajomość pozwala wysyłać zakodowane wiadomości. *Klucz dekodowania*(K_D) - zestaw liczb, których znajomość pozwala wysyłać zakodowane wiadomości.

Na czym polega funkcja dekodująca szyfru Cezara? W miejsce znaku szyfrogramu wpisujemy taki o b niższy.
Co jest kluczem dekodowania w wypadku szyfru Cezara?

Klucze kodowania i dekodowania szyfru Cezara

b jest jedyną liczbą, której znajomość jest potrzebna do kodowania szyfrem Cezara. Dlatego mówimy, że jest *kluczem kodowania*.

Klucze

Klucz kodowania (K_K) - zestaw liczb, których znajomość pozwala wysyłać zakodowane wiadomości. *Klucz dekodowania*(K_D) - zestaw liczb, których znajomość pozwala wysyłać zakodowane wiadomości.

Na czym polega funkcja dekodująca szyfru Cezara? W miejsce znaku szyfrogramu wpisujemy taki o b niższy.
Co jest kluczem dekodowania w wypadku szyfru Cezara? Również b.

Arytmetyka modularna - motywacja

- Do opisu szyfrowania metodą Cezara zwykła arytmetyka nie jest wystarczająca. Jakoś nam wyszło, że $32 + 3 = 2$.

Arytmetyka modularna - motywacja

- Do opisu szyfrowania metodą Cezara zwykła arytmetyka nie jest wystarczająca. Jakoś nam wyszło, że $32 + 3 = 2$.
- Inny przykład: rachuba czasu dobowego.

Arytmetyka modularna - motywacja

- Do opisu szyfrowania metodą Cezara zwykła arytmetyka nie jest wystarczająca. Jakoś nam wyszło, że $32 + 3 = 2$.
- Inny przykład: rachuba czasu dobowego.
- Tak naprawdę, liczymy tylko reszty z dzielenia przez 33 lub przez 24.

Arytmetyka modularna - motywacja

- Do opisu szyfrowania metodą Cezara zwykła arytmetyka nie jest wystarczająca. Jakoś nam wyszło, że $32 + 3 = 2$.
- Inny przykład: rachuba czasu dobowego.
- Tak naprawdę, liczymy tylko reszty z dzielenia przez 33 lub przez 24.
- Takimi działaniami na resztach z dzielenia zajmuje się arytmetyka modularna.

Przystawanie modulo

Przystawanie modulo

Mówimy, że dwie liczby a i b *przystają do siebie modulo* n, jeśli ich różnica $a - b$ jest wielokrotnością n (lub innymi słowy, jeśli liczby te dają tę samą resztę z dzielenia przez n). Zapisujemy to symbolem $a \equiv b(mod\ n)$ lub $a \equiv_n b$.

Np. $8 \equiv 13(mod\ 5)$, bo reszta z dzielenia obu liczb przez 5 to 3.

Przystawanie modulo - własności

Własności modulo

Dla dowolnych a, b, c, d oraz $n > 0$ mamy:

a) $a \equiv_n a$,

b) $a \equiv_n b \Leftrightarrow b \equiv_n a$,

c) jeśli $a \equiv_n b$ i $b \equiv_n c$, to $a \equiv_n c$.

d) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $a + c \equiv_n b + d$,

e) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $a - c \equiv_n b - d$,

f) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $ac \equiv_n bd$.

Przystawanie modulo - własności

Własności modulo

Dla dowolnych a, b, c, d oraz $n > 0$ mamy:

a) $a \equiv_n a$,

b) $a \equiv_n b \Leftrightarrow b \equiv_n a$,

c) jeśli $a \equiv_n b$ i $b \equiv_n c$, to $a \equiv_n c$.

d) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $a + c \equiv_n b + d$,

e) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $a - c \equiv_n b - d$,

f) jeśli $a \equiv_n b$ i $c \equiv_n d$, to $ac \equiv_n bd$.

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$.

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17}$

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17} 8$, $a - b \equiv_{17}$

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17} 8$, $a - b \equiv_{17} 2$, $ab \equiv_{17}$

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17} 8$, $a - b \equiv_{17} 2$, $ab \equiv_{17} 15$.
W skrócie: $5 + 3 \equiv_{17} 8$, $5 - 3 \equiv_{17} 2$, $5 \cdot 3 \equiv_{17} 15$.
Trudniejsze działania:
$3 + 5 \equiv_6$

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17} 8$, $a - b \equiv_{17} 2$, $ab \equiv_{17} 15$.
W skrócie: $5 + 3 \equiv_{17} 8$, $5 - 3 \equiv_{17} 2$, $5 \cdot 3 \equiv_{17} 15$.
Trudniejsze działania:
$3 + 5 \equiv_6 2$, $3 - 5 \equiv_6$

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17} 8$, $a - b \equiv_{17} 2$, $ab \equiv_{17} 15$.
W skrócie: $5 + 3 \equiv_{17} 8$, $5 - 3 \equiv_{17} 2$, $5 \cdot 3 \equiv_{17} 15$.
Trudniejsze działania:
$3 + 5 \equiv_6 2$, $3 - 5 \equiv_6 4$, $3 \cdot 5 \equiv_6$

Działania modulo - przykłady

Dzięki tym własnościom, można zdefiniować tzw. *kongruencje*, czyli działania na resztach z dzielenia, tak samo jak na liczbach.
Niech $a \equiv_{17} 5$ i $b \equiv_{17} 3$. Wtedy $a + b \equiv_{17} 8$, $a - b \equiv_{17} 2$, $ab \equiv_{17} 15$.
W skrócie: $5 + 3 \equiv_{17} 8$, $5 - 3 \equiv_{17} 2$, $5 \cdot 3 \equiv_{17} 15$.
Trudniejsze działania:
$3 + 5 \equiv_6 2$, $3 - 5 \equiv_6 4$, $3 \cdot 5 \equiv_6 3$.

Arytmetyka modularna - przykład $\mathbb{Z}_4$

$\mathbb{Z}_n$: zbiór reszt z dzielenia przez n z działaniami arytmetycznymi modulo n.

Oto „tabliczka dodawania" w zbiorze $\mathbb{Z}_4$:

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Arytmetyka modularna - przykład $\mathbb{Z}_4$

$\mathbb{Z}_n$: zbiór reszt z dzielenia przez n z działaniami arytmetycznymi modulo n.

Oto „tabliczka dodawania” w zbiorze $\mathbb{Z}_4$:

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

Arytmetyka modularna - przykład $\mathbb{Z}_4$

$\mathbb{Z}_n$: zbiór reszt z dzielenia przez n z działaniami arytmetycznymi modulo n.

Oto „tabliczka dodawania" w zbiorze $\mathbb{Z}_4$:

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

I tabliczka mnożenia w tym samym zbiorze:

$\cdot$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Arytmetyka modularna - przykład $\mathbb{Z}_4$

$\mathbb{Z}_n$: zbiór reszt z dzielenia przez n z działaniami arytmetycznymi modulo n.

Oto „tabliczka dodawania" w zbiorze $\mathbb{Z}_4$:

$+$	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

I tabliczka mnożenia w tym samym zbiorze:

$\cdot$	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	0	2
3	0	3	2	1

Odejmowanie łatwo zdefiniować za pomocą dodawania. Dzielenie modulo trzeba wykonywać ostrożnie, bo czasem nie ma sensu (np. $3 : 2$ w $\mathbb{Z}_4$), a czasem może dawać wiele wyników (np. $2 : 2$ w $\mathbb{Z}_4$).

Szyfr Cezara w arytmetyce modularnej

Formalna funkcja kodująca szyfru Cezara to:

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

gdzie b jest dane, a $|J|$ to rozmiar zbioru jednostek szyfrowania.

Szyfr Cezara w arytmetyce modularnej

Formalna funkcja kodująca szyfru Cezara to:

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

gdzie b jest dane, a $|J|$ to rozmiar zbioru jednostek szyfrowania. Funkcja dekodująca tego samego szyfru to:

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

Wady szyfru Cezara

Oczywiście, szyfr Cezara nie nadaje się współcześnie do zakodowania czegokolwiek istotnego:

Wady szyfru Cezara

Oczywiście, szyfr Cezara nie nadaje się współcześnie do zakodowania czegokolwiek istotnego:

- Szyfrogram łatwo rozkodować - dla przykładowej tabeli wystarczy „brutalną siłą” zbadać 33 możliwości (a nawet mniej, bo przesunięcie o 0 jest nonsensowne), a analiza częstości jeszcze sprawę upraszcza. Dlatego nawet kodowanie par lub trójek symboli metodą Cezara nie jest odporne na szybkie złamanie.

Wady szyfru Cezara

Oczywiście, szyfr Cezara nie nadaje się współcześnie do zakodowania czegokolwiek istotnego:

- Szyfrogram łatwo rozkodować - dla przykładowej tabeli wystarczy „brutalną siłą” zbadać 33 możliwości (a nawet mniej, bo przesunięcie o 0 jest nonsensowne), a analiza częstości jeszcze sprawę upraszcza. Dlatego nawet kodowanie par lub trójek symboli metodą Cezara nie jest odporne na szybkie złamanie.
- Gdy generał A wysłał Cezarowi zaszyfrowaną wiadomość, generał B przechwytując kuriera po drodze mógł ją odczytać. Każdy z generałów musiał znać klucz kodujący, by móc wysyłać listy do Cezara, a klucz kodujący był równoważny z dekodującym!

Zasada Kerckhoffsa

Dlatego szyfr Cezara nie spełnia tzw. zasady Kerckhoffsa.

Zasada Kerckhoffsa

Kryptosystem powinien być odporny na złamanie, nawet gdy wszystkie zasady jego działania (poza kluczem dekodowania) są znane.

Zasada Kerckhoffsa

Dlatego szyfr Cezara nie spełnia tzw. zasady Kerckhoffsa.

Zasada Kerckhoffsa

Kryptosystem powinien być odporny na złamanie, nawet gdy wszystkie zasady jego działania (poza kluczem dekodowania) są znane.

Popularne sformułowanie tej zasady, którą powinny spełniać idealne kryptosystemy to: *Wróg zna system, ale i tak nie może zdekodować wiadomości*.

Zasada Kerckhoffsa - uzasadnienie

- Klucze jest dość łatwo zmieniać.

Zasada Kerckhoffsa - uzasadnienie

- Klucze jest dość łatwo zmieniać.
- Zmiana całego kryptosystemu na potrzeby przekazywania każdej kolejnej wiadomości jest niepraktyczna (dla dużej liczby użytkowników np. bank).

Zasada Kerckhoffsa - uzasadnienie

- Klucze jest dość łatwo zmieniać.
- Zmiana całego kryptosystemu na potrzeby przekazywania każdej kolejnej wiadomości jest niepraktyczna (dla dużej liczby użytkowników np. bank).
- Można utajniać algorytmy szyfrowania dla małej grupy osób (np. szyfry wojskowe).

Zasada Kerckhoffsa - uzasadnienie

- Klucze jest dość łatwo zmieniać.
- Zmiana całego kryptosystemu na potrzeby przekazywania każdej kolejnej wiadomości jest niepraktyczna (dla dużej liczby użytkowników np. bank).
- Można utajniać algorytmy szyfrowania dla małej grupy osób (np. szyfry wojskowe).
- *Security by obscurity* - zła praktyka (np. utrudnia poprawianie luk i szacowanie bezpieczeństwa): np. afera *London Oyster*.

Odwracalność funkcji kodującej

Problem: zasada Kerckhoffsa jest niemożliwa do spełnienia...

Odwracalność funkcji kodującej

Problem: zasada Kerckhoffsa jest niemożliwa do spełnienia... Jeśli znamy funkcję f zadaną na zbiorze skończonym, to możemy wyznaczyć (f^{-1}), chociażby wypisując argument po argumencie.

Odwracalność funkcji kodującej

Problem: zasada Kerckhoffsa jest niemożliwa do spełnienia... Jeśli znamy funkcję f zadaną na zbiorze skończonym, to możemy wyznaczyć (f^{-1}), chociażby wypisując argument po argumencie.
1976 Whitfield Diffie i Martin Hellman: wystarczy, by funkcji f nie dało się odwrócić *szybko*.

Odwracalność funkcji kodującej

Problem: zasada Kerckhoffsa jest niemożliwa do spełnienia... Jeśli znamy funkcję f zadaną na zbiorze skończonym, to możemy wyznaczyć (f^{-1}), chociażby wypisując argument po argumencie.
1976 Whitfield Diffie i Martin Hellman: wystarczy, by funkcji f nie dało się odwrócić *szybko*.
Ale czy istnieje prosta procedura („algorytm zapadkowy"), której odwrócenie jest skomplikowane?

Liczby pierwsze - wstęp

Każda liczba $a > 1$ ma **przynajmniej** 2 dzielniki: 1 i samą siebie.

Liczby pierwsze - wstęp

Każda liczba $a > 1$ ma **przynajmniej** 2 dzielniki: 1 i samą siebie. Jednak niektóre z liczb są pod tym względem wyjątkowe:

Liczba pierwsza

Liczba pierwsza to liczba naturalna posiadająca dokładnie 2 różne dzielniki. Liczbę naturalną większą od 1 nazywamy *złożoną*, gdy nie jest pierwsza.

Liczby pierwsze - wstęp

Każda liczba $a > 1$ ma **przynajmniej** 2 dzielniki: 1 i samą siebie. Jednak niektóre z liczb są pod tym względem wyjątkowe:

Liczba pierwsza

Liczba pierwsza to liczba naturalna posiadająca dokładnie 2 różne dzielniki. Liczbę naturalną większą od 1 nazywamy *złożoną*, gdy nie jest pierwsza.

Liczby 0 i 1 nie są uważane ani za pierwsze, ani za złożone.

Liczby pierwsze - wstęp

Każda liczba $a > 1$ ma **przynajmniej** 2 dzielniki: 1 i samą siebie. Jednak niektóre z liczb są pod tym względem wyjątkowe:

Liczba pierwsza

Liczba pierwsza to liczba naturalna posiadająca dokładnie 2 różne dzielniki. Liczbę naturalną większą od 1 nazywamy *złożoną*, gdy nie jest pierwsza.

Liczby 0 i 1 nie są uważane ani za pierwsze, ani za złożone. Zbiór liczb pierwszych oznaczamy $\mathcal{P}$.

Liczby względnie pierwsze

Jeśli $\mathrm{NWD}(a, b) = 1$ to a i b nazywamy *liczbami względnie pierwszymi*. Zapisujemy $a \perp b$.

Liczby pierwsze - znaczenie

Rozkład (faktoryzacja) liczby całkowitej na czynniki pierwsze

Rozkład liczby całkowitej dodatniej na czynniki pierwsze to zapisanie jej w postaci iloczynu $n = p_1^{a_1} p_2^{a_2} \cdot \ldots \cdot p_k^{a_k}$, gdzie $p_1, \ldots p_k$ są różnymi liczbami pierwszymi, a $a_1, \ldots, a_k$ są liczbami całkowitymi dodatnimi.

Liczby pierwsze - znaczenie

Rozkład (faktoryzacja) liczby całkowitej na czynniki pierwsze

Rozkład liczby całkowitej dodatniej na czynniki pierwsze to zapisanie jej w postaci iloczynu $n = p_1^{a_1} p_2^{a_2} \cdot \ldots \cdot p_k^{a_k}$, gdzie $p_1, \ldots p_k$ są różnymi liczbami pierwszymi, a $a_1, \ldots, a_k$ są liczbami całkowitymi dodatnimi.

$$600 = 2^3 \cdot 3 \cdot 5^2 = 5^2 \cdot 2^3 \cdot 3 \ldots$$

Fundamentalne twierdzenie arytmetyki

Każda liczba naturalna $n > 1$ ma jednoznaczny (z dokładnością do kolejności mnożenia) rozkład (czyli faktoryzację) na iloczyn liczb pierwszych.

Liczby pierwsze - znaczenie

Rozkład (faktoryzacja) liczby całkowitej na czynniki pierwsze

Rozkład liczby całkowitej dodatniej na czynniki pierwsze to zapisanie jej w postaci iloczynu $n = p_1^{a_1} p_2^{a_2} \cdot \ldots \cdot p_k^{a_k}$, gdzie $p_1, \ldots p_k$ są różnymi liczbami pierwszymi, a $a_1, \ldots, a_k$ są liczbami całkowitymi dodatnimi.

$$600 = 2^3 \cdot 3 \cdot 5^2 = 5^2 \cdot 2^3 \cdot 3 \ldots$$

Fundamentalne twierdzenie arytmetyki

Każda liczba naturalna $n > 1$ ma jednoznaczny (z dokładnością do kolejności mnożenia) rozkład (czyli faktoryzację) na iloczyn liczb pierwszych.

Fundamentalne twierdzenie arytmetyki

Fundamentalne twierdzenie arytmetyki

Każda liczba naturalna $n > 1$ ma jednoznaczny (z dokładnością do kolejności mnożenia) rozkład (czyli faktoryzację) na iloczyn liczb pierwszych.

Fundamentalne twierdzenie arytmetyki

Fundamentalne twierdzenie arytmetyki

Każda liczba naturalna $n > 1$ ma jednoznaczny (z dokładnością do kolejności mnożenia) rozkład (czyli faktoryzację) na iloczyn liczb pierwszych.

Uwaga! To twierdzenie nie jest konstruktywne!

Fundamentalne twierdzenie arytmetyki

Fundamentalne twierdzenie arytmetyki

Każda liczba naturalna $n > 1$ ma jednoznaczny (z dokładnością do kolejności mnożenia) rozkład (czyli faktoryzację) na iloczyn liczb pierwszych.

Uwaga! To twierdzenie nie jest konstruktywne!
Chociaż mnożenie jest szybkie, to nie ma szybkiego algorytmu faktoryzującego dowolną liczbę naturalną. Najtrudniej rozłożyć iloczyny dwu liczb pierwszych podobnej wielkości.

Fundamentalne twierdzenie arytmetyki

Fundamentalne twierdzenie arytmetyki

Każda liczba naturalna $n > 1$ ma jednoznaczny (z dokładnością do kolejności mnożenia) rozkład (czyli faktoryzację) na iloczyn liczb pierwszych.

Uwaga! To twierdzenie nie jest konstruktywne!
Chociaż mnożenie jest szybkie, to nie ma szybkiego algorytmu faktoryzującego dowolną liczbę naturalną. Najtrudniej rozłożyć iloczyny dwu liczb pierwszych podobnej wielkości.
To jest dokładnie to czego nam było potrzeba: „algorytm zapadkowy”.

Funkcja Eulera - definicja

Funkcja φ Eulera

Funkcja φ Eulera (zwana tocjentem) to $\varphi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ zdefiniowana wzorem:

$$\varphi(n) = \left|\{1 \leqslant a \leqslant n \quad : \quad \mathrm{NWD}(a, n) = 1\}\right|,$$

czyli jest to odwzorowanie przyporządkowujące liczbie n moc zbioru liczb naturalnych dodatnich nie większych od niej i względnie pierwszych z nią.

Funkcja Eulera - definicja

Funkcja φ Eulera

Funkcja φ Eulera (zwana tocjentem) to $\varphi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ zdefiniowana wzorem:

$$\varphi(n) = \left|\{1 \leqslant a \leqslant n \quad : \quad \mathrm{NWD}(a, n) = 1\}\right|,$$

czyli jest to odwzorowanie przyporządkowujące liczbie n moc zbioru liczb naturalnych dodatnich nie większych od niej i względnie pierwszych z nią.

Przykładowo obliczmy $\varphi(6)$.

Funkcja Eulera - definicja

Funkcja φ Eulera

Funkcja φ Eulera (zwana tocjentem) to $\varphi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ zdefiniowana wzorem:

$$\varphi(n) = \left|\{1 \leqslant a \leqslant n \quad : \quad \mathrm{NWD}(a, n) = 1\}\right|,$$

czyli jest to odwzorowanie przyporządkowujące liczbie n moc zbioru liczb naturalnych dodatnich nie większych od niej i względnie pierwszych z nią.

Przykładowo obliczmy $\varphi(6)$. Liczby nie większe od niej i względnie pierwsze z nią to

Funkcja Eulera - definicja

Funkcja φ Eulera

Funkcja φ Eulera (zwana tocjentem) to $\varphi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ zdefiniowana wzorem:

$$\varphi(n) = \left|\{1 \leqslant a \leqslant n \quad : \quad \mathrm{NWD}(a, n) = 1\}\right|,$$

czyli jest to odwzorowanie przyporządkowujące liczbie n moc zbioru liczb naturalnych dodatnich nie większych od niej i względnie pierwszych z nią.

Przykładowo obliczmy $\varphi(6)$. Liczby nie większe od niej i względnie pierwsze z nią to 1 i 5, więc $\varphi(6) =$

Funkcja Eulera - definicja

Funkcja φ Eulera

Funkcja φ Eulera (zwana tocjentem) to $\varphi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ zdefiniowana wzorem:

$$\varphi(n) = \left|\{1 \leqslant a \leqslant n \quad : \quad \mathrm{NWD}(a, n) = 1\}\right|,$$

czyli jest to odwzorowanie przyporządkowujące liczbie n moc zbioru liczb naturalnych dodatnich nie większych od niej i względnie pierwszych z nią.

Przykładowo obliczmy $\varphi(6)$. Liczby nie większe od niej i względnie pierwsze z nią to 1 i 5, więc $\varphi(6) = 2$.

Funkcja Eulera - definicja

Funkcja φ Eulera

Funkcja φ Eulera (zwana tocjentem) to $\varphi : \mathbb{N} \setminus \{0\} \longrightarrow \mathbb{N}$ zdefiniowana wzorem:

$$\varphi(n) = \left|\{1 \leqslant a \leqslant n \quad : \quad \mathrm{NWD}(a, n) = 1\}\right|,$$

czyli jest to odwzorowanie przyporządkowujące liczbie n moc zbioru liczb naturalnych dodatnich nie większych od niej i względnie pierwszych z nią.

Przykładowo obliczmy $\varphi(6)$. Liczby nie większe od niej i względnie pierwsze z nią to 1 i 5, więc $\varphi(6) = 2$.
Oczywiście, nie jest to zbyt wygodny sposób obliczania wartości funkcji φ.

Funkcja Eulera - twierdzenia

Funkcja φ Eulera dla liczb pierwszych

Dla dowolnej liczby pierwszej p zachodzą związki:

a) $\varphi(p) = p - 1$

b) $\varphi(p^k) = p^k(1 - \frac{1}{p})$.

Funkcja Eulera - twierdzenia

Funkcja φ Eulera dla liczb pierwszych

Dla dowolnej liczby pierwszej p zachodzą związki:
a) $\varphi(p) = p - 1$
b) $\varphi(p^k) = p^k(1 - \frac{1}{p})$.

Funkcja φ Eulera dla iloczynów liczb względnie pierwszych

Dla dowolnych dwóch dodatnich liczb względnie pierwszych m i n zachodzi:
$\varphi(mn) = \varphi(m)\varphi(n)$.

Funkcja Eulera - twierdzenia

Funkcja φ Eulera dla liczb pierwszych

Dla dowolnej liczby pierwszej p zachodzą związki:
a) $\varphi(p) = p - 1$
b) $\varphi(p^k) = p^k(1 - \frac{1}{p})$.

Funkcja φ Eulera dla iloczynów liczb względnie pierwszych

Dla dowolnych dwóch dodatnich liczb względnie pierwszych m i n zachodzi:
$\varphi(mn) = \varphi(m)\varphi(n)$.

Wniosek - potrafimy policzyć wartość funkcji Eulera dla każdej liczby, której rozkład na czynniki pierwsze znamy.

Funkcja Eulera - twierdzenia

Funkcja φ Eulera dla liczb pierwszych

Dla dowolnej liczby pierwszej p zachodzą związki:
a) $\varphi(p) = p - 1$
b) $\varphi(p^k) = p^k(1 - \frac{1}{p})$.

Funkcja φ Eulera dla iloczynów liczb względnie pierwszych

Dla dowolnych dwóch dodatnich liczb względnie pierwszych m i n zachodzi:
$\varphi(mn) = \varphi(m)\varphi(n)$.

Wniosek - potrafimy policzyć wartość funkcji Eulera dla każdej liczby, której rozkład na czynniki pierwsze znamy. Ale tylko takiej!

Funkcja Eulera - przykład

Zadanie

Obliczyć $\varphi(600)$

Funkcja Eulera - przykład

Zadanie

Obliczyć $\varphi(600)$

Łatwo zauważyć, że $600 = 2^3 \cdot 3 \cdot 5^2$.

Funkcja Eulera - przykład

Zadanie

Obliczyć $\varphi(600)$

Łatwo zauważyć, że $600 = 2^3 \cdot 3 \cdot 5^2$. Dodatkowo, oczywiście 2^3, 3 i 5^2 są parami względnie pierwsze, więc:

$$\varphi(600) = \varphi(2^3 \cdot 3 \cdot 5^2)$$

Funkcja Eulera - przykład

Zadanie

Obliczyć $\varphi(600)$

Łatwo zauważyć, że $600 = 2^3 \cdot 3 \cdot 5^2$. Dodatkowo, oczywiście 2^3, 3 i 5^2 są parami względnie pierwsze, więc:

$$\varphi(600) = \varphi(2^3 \cdot 3 \cdot 5^2) = \varphi(2^3) \cdot \varphi(3) \cdot \varphi(5^2)$$

Funkcja Eulera - przykład

Zadanie

Obliczyć $\varphi(600)$

Łatwo zauważyć, że $600 = 2^3 \cdot 3 \cdot 5^2$. Dodatkowo, oczywiście 2^3, 3 i 5^2 są parami względnie pierwsze, więc:

$$\varphi(600) = \varphi(2^3 \cdot 3 \cdot 5^2) = \varphi(2^3) \cdot \varphi(3) \cdot \varphi(5^2) = 2^3(1-\frac{1}{2}) \cdot 2 \cdot 5^2(1-\frac{1}{5})$$

Funkcja Eulera - przykład

Zadanie

Obliczyć $\varphi(600)$

Łatwo zauważyć, że $600 = 2^3 \cdot 3 \cdot 5^2$. Dodatkowo, oczywiście 2^3, 3 i 5^2 są parami względnie pierwsze, więc:

$$\varphi(600) = \varphi(2^3 \cdot 3 \cdot 5^2) = \varphi(2^3) \cdot \varphi(3) \cdot \varphi(5^2) = 2^3(1-\frac{1}{2}) \cdot 2 \cdot 5^2(1-\frac{1}{5}) = 160.$$

Co można zrobić szybko?

Teoria liczb gwarantuje, że da się relatywnie *szybko* wykonać:

Co można zrobić szybko?

Teoria liczb gwarantuje, że da się relatywnie *szybko* wykonać:

- mnożenie dwóch liczb;

Co można zrobić szybko?

Teoria liczb gwarantuje, że da się relatywnie *szybko* wykonać:

- mnożenie dwóch liczb;
- sprawdzenie, czy dana liczba jest pierwsza;

Co można zrobić szybko?

Teoria liczb gwarantuje, że da się relatywnie *szybko* wykonać:

- mnożenie dwóch liczb;
- sprawdzenie, czy dana liczba jest pierwsza;
- dla danych liczb a i n wyznaczenie takiej liczby, że $ab \equiv 1 (\mod \varphi(n))$ (rozszerzony algorytm Euklidesa);

Co można zrobić szybko?

Teoria liczb gwarantuje, że da się relatywnie *szybko* wykonać:

- mnożenie dwóch liczb;
- sprawdzenie, czy dana liczba jest pierwsza;
- dla danych liczb a i n wyznaczenie takiej liczby, że $ab \equiv 1 (\mod \varphi(n))$ (rozszerzony algorytm Euklidesa);
- potęgowanie na resztach z dzielenia (potęgowanie „przez kwadraty").

Czego nie da się zrobić szybko?

Teoria liczb gwarantuje, że **nie da się** relatywnie *szybko* wykonać:

Czego nie da się zrobić szybko?

Teoria liczb gwarantuje, że **nie da się** relatywnie *szybko* wykonać:

- rozkładu dużej liczby na czynniki pierwsze (zwłaszcza w wypadku iloczynu dwóch dużych liczb pierwszych);

Czego nie da się zrobić szybko?

Teoria liczb gwarantuje, że **nie da się** relatywnie *szybko* wykonać:

- rozkładu dużej liczby na czynniki pierwsze (zwłaszcza w wypadku iloczynu dwóch dużych liczb pierwszych);
- obliczenia wartości funkcji Eulera dla dużych argumentów.

Kryptosystem z kluczem publicznym

Kryptosystem z kluczem publicznym

Kryptosystem z kluczem publicznym to system złożony ze zbioru jednostek tekstu, funkcji kodującej oraz kluczy kodowania i dekodowania takich, że:

a) Klucz kodowania jest powszechnie znany, a klucz dekodowania jest tajny.

b) Klucz kodowania pozwala na szybkie zaszyfrowanie jednostki tekstu jawnego, czyli obliczanie wartości funkcji kodującej.

c) Klucz dekodowania pozwala na szybkie zdeszyfrowanie jednostki szyfrogramu, czyli obliczanie wartości funkcji dekodującej.

d) Znajomość klucza kodowania nie pozwala (bez znajomości klucza dekodowania) na szybkie dekodowanie jednostki szyfrogramu.

Kryptosystem z kluczem publicznym

Kryptosystem z kluczem publicznym

Kryptosystem z kluczem publicznym to system złożony ze zbioru jednostek tekstu, funkcji kodującej oraz kluczy kodowania i dekodowania takich, że:
a) Klucz kodowania jest powszechnie znany, a klucz dekodowania jest tajny.
b) Klucz kodowania pozwala na szybkie zaszyfrowanie jednostki tekstu jawnego, czyli obliczanie wartości funkcji kodującej.
c) Klucz dekodowania pozwala na szybkie zdeszyfrowanie jednostki szyfrogramu, czyli obliczanie wartości funkcji dekodującej.
d) Znajomość klucza kodowania nie pozwala (bez znajomości klucza dekodowania) na szybkie dekodowanie jednostki szyfrogramu.

Przykład: algorytm RSA (1977, od nazwisk pomysłodawców: Rivesta, Shamira i Adlemana, obecnie w domenie publicznej).

Algorytm RSA - część I

Algorytm RSA

Algorytm RSA - część I

Algorytm RSA

- I. Wybierz dwie różne, bardzo duże liczby pierwsze (p i q). Im większe, tym trudniej złamać kod, ale też dłużej trwa proces kodowania i dekodowania.

Algorytm RSA - część I

Algorytm RSA

- I. Wybierz dwie różne, bardzo duże liczby pierwsze (p i q). Im większe, tym trudniej złamać kod, ale też dłużej trwa proces kodowania i dekodowania.
- II. Niech $n = pq$. Wtedy $\varphi(n) = \varphi(p) \cdot \varphi(q) = (p-1)(q-1)$.

Algorytm RSA - część I

Algorytm RSA

- I. Wybierz dwie różne, bardzo duże liczby pierwsze (p i q). Im większe, tym trudniej złamać kod, ale też dłużej trwa proces kodowania i dekodowania.
- II. Niech $n = pq$. Wtedy $\varphi(n) = \varphi(p) \cdot \varphi(q) = (p-1)(q-1)$.
- III. Wybieramy liczbę e względnie pierwszą z $\varphi(n)$. Znajdujemy d takie, że $e \cdot d \equiv 1(\mod \varphi(n))$.
- IV. Klucz kodowania $K_K = (n, e)$.
  Klucz dekodowania $K_D = (n, d)$.

Algorytm RSA - część I

Algorytm RSA

- I. Wybierz dwie różne, bardzo duże liczby pierwsze (p i q). Im większe, tym trudniej złamać kod, ale też dłużej trwa proces kodowania i dekodowania.
- II. Niech $n = pq$. Wtedy $\varphi(n) = \varphi(p) \cdot \varphi(q) = (p-1)(q-1)$.
- III. Wybieramy liczbę e względnie pierwszą z $\varphi(n)$. Znajdujemy d takie, że $e \cdot d \equiv 1(\mod \varphi(n))$.
- IV. Klucz kodowania $K_K = (n, e)$.
  Klucz dekodowania $K_D = (n, d)$.
- V. Jednostki tekstu $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$.
  Funkcja kodująca: $f(P) = P^e \mod n$.

Algorytm RSA - część I

Algorytm RSA

- I. Wybierz dwie różne, bardzo duże liczby pierwsze (p i q). Im większe, tym trudniej złamać kod, ale też dłużej trwa proces kodowania i dekodowania.
- II. Niech $n = pq$. Wtedy $\varphi(n) = \varphi(p) \cdot \varphi(q) = (p-1)(q-1)$.
- III. Wybieramy liczbę e względnie pierwszą z $\varphi(n)$. Znajdujemy d takie, że $e \cdot d \equiv 1(\mod \varphi(n))$.
- IV. Klucz kodowania $K_K = (n, e)$.
  Klucz dekodowania $K_D = (n, d)$.
- V. Jednostki tekstu $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$.
  Funkcja kodująca: $f(P) = P^e \mod n$.
- VI. Funkcja dekodująca: $f^{-1}(C) = C^d \mod n$.

Algorytm RSA - uwagi

- f i f^{-1} w RSA faktycznie są operacjami odwrotnymi.

Algorytm RSA - uwagi

- f i f^{-1} w RSA faktycznie są operacjami odwrotnymi.
- Warto przygotować zbiór jednostek do kodowania (tzw. padding), dodać szum informacyjny - pozamatematyczne elementy kryptologii.

Algorytm RSA - uwagi

- f i f^{-1} w RSA faktycznie są operacjami odwrotnymi.
- Warto przygotować zbiór jednostek do kodowania (tzw. padding), dodać szum informacyjny - pozamatematyczne elementy kryptologii.
- Kryptosystem RSA pozostaje bezpieczny, o ile klucze są odpowiednio długie. Obecnie używa się kluczy 1024- lub 2048-bitowych.

Algorytm RSA - uwagi

- f i f^{-1} w RSA faktycznie są operacjami odwrotnymi.
- Warto przygotować zbiór jednostek do kodowania (tzw. padding), dodać szum informacyjny - pozamatematyczne elementy kryptologii.
- Kryptosystem RSA pozostaje bezpieczny, o ile klucze są odpowiednio długie. Obecnie używa się kluczy 1024- lub 2048-bitowych.
- Klucze ponad 800-bitowe nie są do dziś złamane. Klucze o rozmiarze 300 bitów lub mniejszym da się złamać przy pomocy zwyczajnych komputerów i darmowego oprogramowania w kilka godzin.

Algorytm RSA - przykład

Spróbujmy algorytmem RSA zakodować zdanie: Alea iacta est!

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

Algorytm RSA - przykład

Spróbujmy algorytmem RSA zakodować zdanie: Alea iacta est!

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

Jak widać, $n = 33$, zatem

$\varphi(n) = \varphi(3 \cdot 11) = \varphi(3) \cdot \varphi(11) = 2 \cdot 10 = 20.$

Algorytm RSA - przykład

Spróbujmy algorytmem RSA zakodować zdanie: Alea iacta est!

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

Jak widać, $n = 33$, zatem
$\varphi(n) = \varphi(3 \cdot 11) = \varphi(3) \cdot \varphi(11) = 2 \cdot 10 = 20$. Do klucza publicznego musimy wybrać e względnie pierwsze z $\varphi(n) = 20$.

Algorytm RSA - przykład

Spróbujmy algorytmem RSA zakodować zdanie: Alea iacta est!

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

Jak widać, $n = 33$, zatem
$\varphi(n) = \varphi(3 \cdot 11) = \varphi(3) \cdot \varphi(11) = 2 \cdot 10 = 20$. Do klucza publicznego musimy wybrać e względnie pierwsze z $\varphi(n) = 20$. Niech $e = 7$ - czyli cały klucz publiczny to $(33, 7)$.

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Pierwszą literą, którą mamy zakodować, jest A. Odpowiada jej liczba 3.

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Pierwszą literą, którą mamy zakodować, jest A. Odpowiada jej liczba 3. $3^7 = 2187$.

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Pierwszą literą, którą mamy zakodować, jest A. Odpowiada jej liczba 3. $3^7 = 2187$. Np. korzystając z kalkulatora obliczamy, że $2187 : 33 \approx 66,27$

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Pierwszą literą, którą mamy zakodować, jest A. Odpowiada jej liczba 3. $3^7 = 2187$. Np. korzystając z kalkulatora obliczamy, że $2187 : 33 \approx 66,27$, a $2187 - 66 \cdot 33 = 2187 - 2178 = 9$,

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Pierwszą literą, którą mamy zakodować, jest A. Odpowiada jej liczba 3. $3^7 = 2187$. Np. korzystając z kalkulatora obliczamy, że $2187 : 33 \approx 66,27$, a $2187 - 66 \cdot 33 = 2187 - 2178 = 9$, zatem $f(3) = 2187 \mod 33 = 9$,

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Pierwszą literą, którą mamy zakodować, jest A. Odpowiada jej liczba 3. $3^7 = 2187$. Np. korzystając z kalkulatora obliczamy, że $2187 : 33 \approx 66,27$, a $2187 - 66 \cdot 33 = 2187 - 2178 = 9$, zatem $f(3) = 2187 \mod 33 = 9$, a liczbie 9 odpowiada w naszej tabelce G, więc litera A zostanie zakodowana jako G.

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest L. Odpowiada jej liczba 14.

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest *L*. Odpowiada jej liczba 14.
Potęgowanie, które musimy wykonać można sobie ułatwić „potęgując przez kwadraty" (komputery też tak robią) i sprytnie wykorzystując własności kongruencji:

$$14^7 =$$

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest *L*. Odpowiada jej liczba 14.
Potęgowanie, które musimy wykonać można sobie ułatwić „potęgując przez kwadraty" (komputery też tak robią) i sprytnie wykorzystując własności kongruencji:

$$14^7 = (14^2)^3 \cdot 14 \equiv_{33}$$

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest *L*. Odpowiada jej liczba 14.
Potęgowanie, które musimy wykonać można sobie ułatwić „potęgując przez kwadraty" (komputery też tak robią) i sprytnie wykorzystując własności kongruencji:

$$14^7 = (14^2)^3 \cdot 14 \equiv_{33} 31^3 \cdot 14 =$$

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest L. Odpowiada jej liczba 14.
Potęgowanie, które musimy wykonać można sobie ułatwić „potęgując przez kwadraty" (komputery też tak robią) i sprytnie wykorzystując własności kongruencji:

$$14^7 = (14^2)^3 \cdot 14 \equiv_{33} 31^3 \cdot 14 = 31^2 \cdot (31 \cdot 14) \equiv_{33}$$

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest L. Odpowiada jej liczba 14.
Potęgowanie, które musimy wykonać można sobie ułatwić „potęgując przez kwadraty" (komputery też tak robią) i sprytnie wykorzystując własności kongruencji:

$$14^7 = (14^2)^3 \cdot 14 \equiv_{33} 31^3 \cdot 14 = 31^2 \cdot (31 \cdot 14) \equiv_{33}$$

$$\equiv_{33} (-2)^2 \cdot ((-2) \cdot 14) =$$

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest L. Odpowiada jej liczba 14.
Potęgowanie, które musimy wykonać można sobie ułatwić „potęgując przez kwadraty" (komputery też tak robią) i sprytnie wykorzystując własności kongruencji:

$$14^7 = (14^2)^3 \cdot 14 \equiv_{33} 31^3 \cdot 14 = 31^2 \cdot (31 \cdot 14) \equiv_{33}$$

$$\equiv_{33} (-2)^2 \cdot ((-2) \cdot 14) = 4 \cdot (-28) \equiv_{33}$$

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!
Drugą literą, którą mamy zakodować, jest L. Odpowiada jej liczba 14. Potęgowanie, które musimy wykonać można sobie ułatwić „potęgując przez kwadraty" (komputery też tak robią) i sprytnie wykorzystując własności kongruencji:

$$14^7 = (14^2)^3 \cdot 14 \equiv_{33} 31^3 \cdot 14 = 31^2 \cdot (31 \cdot 14) \equiv_{33}$$

$$\equiv_{33} (-2)^2 \cdot ((-2) \cdot 14) = 4 \cdot (-28) \equiv_{33} 4 \cdot 5 = 20.$$

Liczbie 20 przyporządkowana jest litera R, więc L będzie zakodowane jako R.

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Analogicznie kodujemy kolejne litery otrzymując (jako ćwiczenie - sprawdzić):

grzgoigltgozst!

Algorytm RSA - przykład

Klucz publiczny $(33, 7)$. Tekst jawny: Alea iacta est!

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

Analogicznie kodujemy kolejne litery otrzymując (jako ćwiczenie - sprawdzić):

grzgoigltgozst!

Niektóre znaki (np. „i” „„s”, „t”, „!”) kodują same siebie - jest to raczej nieuniknione przy tak małych liczbach tworzących klucz publiczny (co jest kolejnym słabym punktem wyboru małych liczb w algorytmie RSA).

Algorytm RSA - dekodowanie

Klucz publiczny $(33, 7)$. Szyfrogram: grzgoigltgozst!

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

Algorytm RSA - dekodowanie

Klucz publiczny $(33, 7)$. Szyfrogram: grzgoigltgozst!

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

Spróbujemy złamać szyfr przez wygenerowanie klucza dekodującego.

Algorytm RSA - dekodowanie

Klucz publiczny $(33, 7)$. Szyfrogram: grzgoigltgozst!

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

Spróbujemy złamać szyfr przez wygenerowanie klucza dekodującego. Oczywiście $n = 33$, ale musimy odtworzyć d (da się tylko dlatego, że mamy małą liczbę n).

Algorytm RSA - dekodowanie

Klucz publiczny $(33, 7)$. Szyfrogram: grzgoigltgozst!

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

Spróbujemy złamać szyfr przez wygenerowanie klucza dekodującego. Oczywiście $n = 33$, ale musimy odtworzyć d (da się tylko dlatego, że mamy małą liczbę n). Obliczamy $\varphi(33) = 20$.

Algorytm RSA - dekodowanie

Klucz publiczny $(33, 7)$. Szyfrogram: grzgoigltgozst!

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

Spróbujemy złamać szyfr przez wygenerowanie klucza dekodującego. Oczywiście $n = 33$, ale musimy odtworzyć d (da się tylko dlatego, że mamy małą liczbę n). Obliczamy $\varphi(33) = 20$. $7d \equiv 1(\mod 20)$.

Algorytm RSA - dekodowanie

Klucz publiczny $(33, 7)$. Szyfrogram: grzgoigltgozst!

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

Spróbujemy złamać szyfr przez wygenerowanie klucza dekodującego. Oczywiście $n = 33$, ale musimy odtworzyć d (da się tylko dlatego, że mamy małą liczbę n). Obliczamy $\varphi(33) = 20$. $7d \equiv 1(\mod 20)$. Oczywiście, pasuje $d = 3$, bo $7 \cdot 3 = 1 \cdot 20 + 1$.

Algorytm RSA - dekodowanie

Klucz publiczny $(33, 7)$. Szyfrogram: grzgoigltgozst!

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

Spróbujemy złamać szyfr przez wygenerowanie klucza dekodującego. Oczywiście $n = 33$, ale musimy odtworzyć d (da się tylko dlatego, że mamy małą liczbę n). Obliczamy $\varphi(33) = 20$. $7d \equiv 1 \pmod{20}$. Oczywiście, pasuje $d = 3$, bo $7 \cdot 3 = 1 \cdot 20 + 1$. Zatem klucz prywatny, to $(33, 3)$, a funkcją dekodującą jest $f^{-1}(C) = C^3 \mod n$.

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Powiedzmy, że chcemy odkodować „z”, występujące w szyfrogramie. Liczba odpowiadająca „z” to 28. Tak jak poprzednio, obliczamy

$$28^3 \equiv_{33}$$

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Powiedzmy, że chcemy odkodować „z”, występujące w szyfrogramie. Liczba odpowiadająca „z” to 28. Tak jak poprzednio, obliczamy

$$28^3 \equiv_{33} (-5)^3 \equiv_{33}$$

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Powiedzmy, że chcemy odkodować „z”, występujące w szyfrogramie. Liczba odpowiadająca „z” to 28. Tak jak poprzednio, obliczamy

$$28^3 \equiv_{33} (-5)^3 \equiv_{33} 25 \cdot (-5) \equiv_{33}$$

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Powiedzmy, że chcemy odkodować „z”, występujące w szyfrogramie. Liczba odpowiadająca „z” to 28. Tak jak poprzednio, obliczamy

$$28^3 \equiv_{33} (-5)^3 \equiv_{33} 25 \cdot (-5) \equiv_{33} (-8) \cdot (-5) \equiv_{33}$$

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Powiedzmy, że chcemy odkodować „z”, występujące w szyfrogramie. Liczba odpowiadająca „z” to 28. Tak jak poprzednio, obliczamy

$$28^3 \equiv_{33} (-5)^3 \equiv_{33} 25 \cdot (-5) \equiv_{33} (-8) \cdot (-5) \equiv_{33} 40 \equiv_{33}$$

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Powiedzmy, że chcemy odkodować „z”, występujące w szyfrogramie. Liczba odpowiadająca „z” to 28. Tak jak poprzednio, obliczamy

$$28^3 \equiv_{33} (-5)^3 \equiv_{33} 25 \cdot (-5) \equiv_{33} (-8) \cdot (-5) \equiv_{33} 40 \equiv_{33} 7.$$

Algorytm RSA - dekodowanie

Klucz prywatny $(33, 3)$. Szyfrogram: grzgoigltgozst!

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

Powiedzmy, że chcemy odkodować „z”, występujące w szyfrogramie. Liczba odpowiadająca „z” to 28. Tak jak poprzednio, obliczamy

$$28^3 \equiv_{33} (-5)^3 \equiv_{33} 25 \cdot (-5) \equiv_{33} (-8) \cdot (-5) \equiv_{33} 40 \equiv_{33} 7.$$

7 odpowiada literze „e”, więc wiemy, że „z” możemy odkodować jako „e”. Resztę wiadomości dekodujemy analogicznie.

Inne algorytmy kryptograficzne

- Protokół Rabina;

Inne algorytmy kryptograficzne

- Protokół Rabina;
- Protokół Diffiego-Hellmana i algorytm ElGamal;

Inne algorytmy kryptograficzne

- Protokół Rabina;
- Protokół Diffiego-Hellmana i algorytm ElGamal;
- Kryptografia krzywych eliptycznych (Koblitz, Miller 1985);

Inne algorytmy kryptograficzne

- Protokół Rabina;
- Protokół Diffiego-Hellmana i algorytm ElGamal;
- Kryptografia krzywych eliptycznych (Koblitz, Miller 1985);
- Gdy obie strony komunikacji mają możliwość wcześniejszego uzgodnienia wspólnego klucza używanego wyłącznie do kontaktów między nimi, używane są algorytmy *kryptografii symetrycznej* np. AES (Rijndael), Twofish, Serpent, dawniej DES.