Szyfrowanie

Szyfrowanie

Sposób na zakodowanie danych w taki sposób, aby stały się nieczytelne. Proces ten jest jednak **odwracalny**. Metoda ta służy m. in. do poufnego przekazywania informacji.

Algorytmy szyfruj?ce

Algorytm szyfrujący to sposób w jaki zamieniamy dane na nieczytelne. Jest wiele sposobów na szyfrowanie - czyli jest wiele algorytmów.

Jeśli chodzi o kategorie do których się je da zaliczyć w tej chwili najpopularniejszy podział to podział na szyfrowanie symetryczne i asymetryczne.

Szyfrowanie symetryczne.

- ten sam klucz jest wykorzystywany do szyfrowania i deszyfrowania danych.

Czyli obie strony (nadawca i odbiorca wiadomości) dzielą się tym samym sposobem ("hasłem" / "kluczem") umożliwiającym zakodowanie i odkodowanie wiadomości.

Pojawia się więc problem - jak przekazać bezpiecznie ten klucz? Zwłaszcza w sytuacji gdy nadawca i odbiorca nie mogą się jakoś "spotkać" w bezpiecznym miejscu, żeby podzielić się tym kluczem - np. jedna strona jest gdzieś w Polsce a druga np. w Brazylii.

Mimo tego problemu szyfrowanie tego rodzaju jest stosowane m.in. ze względu na szybkość (sprawnie działa nawet na urządzeniach o bardzo ograniczonych zasobach).

Szyfrowanie symetryczne

Bob

Bob chce się
skomunikować z
Alice...
ale chce żeby nikt
inny nie odczytał tej
wiadomości

Alice

klucz
symetryczny

Bob

wysyła jej więc klucz
do szyfrowania
i odszyfrowywania
wiadomości

Alice

dokument
zaszyfrowany
kluczem
symetrycznym

Bob

szyfruje więc wiadomość
kluczem i wysyła do Alice

Alice

Alice używa klucza
do odszyfrowania wiadomości

klucz
symetryczny

Szyfrowanie asymetryczne

- wykorzystuje dwa oddzielne klucze, jeden służy do szyfrowania danych, drugi do odszyfrowania

Ja lubię myśleć o tym w ten sposób - mamy kłódkę i klucz. Kłódkę udostępniamy publicznie, każdy może sobie ją pobrać i zaszyfrować dane z jej pomocą. My jednak mamy klucz, którym się nie dzielimy - to nasz sekret (klucz prywatny). Tylko z jego pomocą można "otworzyć" (odszyfrować) dane zaszyfrowane z pomocą "kłódki".

Dzięki temu pozbywamy się problemu z przekazaniem sposobu odszyfrowania wiadomości. De facto nie robimy tego - przekazujemy jedynie sposób na zakodowanie.

Jeśli komunikacja ma zachodzić dwukierunkowo obie strony wysyłają sobie nawzajem "kłódki" (tzw. klucze publiczne).

Ta metoda ma jednak dość poważną wadę - dużo bardziej niż algorytmy symetryczne obciąża nasze komputery.

Zwróćcie jednak uwagę, że możemy wykorzystać wolne szyfrowanie asymetryczne, żeby podzielić się z kimś szybkim kluczem symetrycznym ;) .

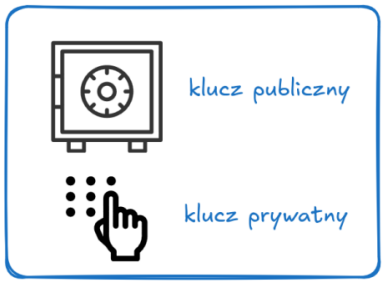
Szyfrowanie asymetryczne

Bob

Bob chce się
skomunikować z
Alice...
ale chce żeby nikt
inny nie odczytał tej
wiadomości

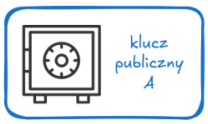
Alice

Alice tworzy u siebie
parę kluczy -
jeden służy do szyfrowania,
drugi do odszyfrowania wiadomości



para kluczy
asymetrycznych

Bob



Alice wysyła do Boba
klucz do szyfrowania
(klucz publiczny)

Alice

Bob

dokument
zaszyfrowany
kluczem
publicznym A

Bob wysyła do Alice
wiadomość zaszyfrowaną
z pomocą jej klucza
publicznego.
Zwróć uwagę, że od tego momentu
Bob nie ma możliwości odszyfrowania
treści - klucz do tego celu ma Alice

Alice

Alice używa swojego
klucza prywatnego
do odszyfrowania wiadomości
zakodowanej jej kluczem publicznym

Funkcja skrótu (hash)

Nie należy mylić szyfrowania z hashowaniem. Funkcja skrótu tworzy nam "bełkot" na podstawie jakiś danych wejściowych. Proces ten jest (a właściwie powinien być) NIEODWRACALNY.

Jednak zawsze jesteśmy w stanie stwierdzić, że dany hash powstał na podstawie konkretnych danych wejściowych. W najprostszej postaci - te same dane wejściowe zawsze dadzą nam ten sam hash.

Co może wydawać się dziwne - po co nam coś takiego?

To super przydatne narzędzie. Umożliwia np. bezpieczne przechowywanie haseł w bazach danych. Nie przechowujemy de facto haseł tylko ich hashe - przez co w przypadku wycieku strona atakująca nie dostaje haseł.

Wykorzystując hashe jesteśmy w stanie stwierdzić czy dany pliki / wiadomości zostały zmodyfikowane - bo zmodyfikowane dane dadzą inny hash.

Programy antywirusowe nie muszą przechowywać kodu całego wirusa, żeby go namierzyć - wystarczy, że mają jego hash i szukają jego wystąpień.

Słowniki i zbiory korzystają z funkcji skrótu do wskazania miejsca gdzie można znaleźć nasze dane (to dzięki temu słowniki są szybsze od list przy dużej ilości elementów).

Szyfr Cezara

To jedna z najprostszych technik szyfrowania - do tego idealnie nadająca się do przećwiczenia pętli.

Polega ona na podmianie każdej litery tekstu jawnego (niezaszyfrowanego) inną, oddaloną o stałą liczbę pozycji w alfabecie. Liczba ta staje się kluczem do wiadomości.

np.

```
Alfabet: AĄBCĆDEĘFGHIJKLŁMNŃOÓPRSŚTUWYZŹŻ
```

Chcemy zaszyfrować imię `ADA`. Jako klucz przyjmijmy 3 - co oznacza, że każdą literę podmieniamy na odpowiednik po prawej stronie oddalony o 3 pozycje.

Czyli dla litery `A` będzie to `C`. Stąd też całość będzie wyglądała następująco:

```
Wiadomość: ADA
Klucz: 3
```

```
Zaszyfrowana: CFC
```

Co robić w przypadku wyjścia poza naszą tablicę znaków? Jak znowu przeskoczyć na początek?

Z pomocą przyjdzie nam modulo. Aby z jego pomocą wskazać indeks zaszyfrowanego znaku w naszym alfabecie możemy skorzystać z formuły:

```
indekszaszyfrowany = (indeksznaku + klucz) % len(ZNAKI)
```

Odno?niki

„3.12.5 Documentation”. Dostęp 18 sierpień 2024. https://docs.python.org/3/.

Pengelly, James, i Gareth Marchant. The Official CompTIA Security+ Student Guide (Exam SY0-701). CompTIA, Inc, 2023.

„Szyfr Cezara”. W *Wikipedia, wolna encyklopedia*, 13 czerwiec 2024. https://pl.wikipedia.org/w/index.php?title=Szyfr_Cezara&oldid=74000247.

Wersja #2
Utworzono 2024-09-23 14:27:54 UTC przez Przemek Jeske
Zaktualizowano 2025-08-03 13:01:22 UTC przez Przemek Jeske