#17

Syahrul Imardi, MT

STMIK Amik Riau

MATAKULIAH

KEAMANAN PERANGKAT LUNAK

One-Time Pad

P17

STMIK
Amik Riau

MATAKULIAH
KEAMANAN PERANGKAT LUNAK

Syahrul Imardi, MT

P17 : *One-Time Pad*

One-Time Pad, *Cipher* yang Tidak Dapat Dipecahkan (*Unbreakable Cipher*)

Pendahuluan

- *Unbreakable cipher* merupakan klaim yang dibuat oleh kriptografer terhadap algoritma kriptografi yang dirancangnya.

- Namun, kebanyakan algoritma yang sudah pernah dibuat orang adalah *breakable cipher*.

- *Caesar Cipher*, *Vigenere Cipher* , *Playfair Cipher*, *Enigma Cipher*, *Hill Cipher*, dll sudah kadaluarsa karena *breakable cipher*.

- Apakah *unbreakable cipher* memang benar-benar ada?

  Jawaban: ada

- Apa syarat sebuah *cipher* disebut *unbreakable cipher*?

  Jawaban:

  1. Kunci harus benar-benar acak (*trully random*).
  2. Panjang kunci = panjang plainteks

- Acak: tidak dapat diprediksi nilainya dan tidak dapat diulang

- Akibat 1 dan 2: plainteks yang sama tidak selalu menghasilkan cipherteks yang sama

One-Time Pad (OTP)

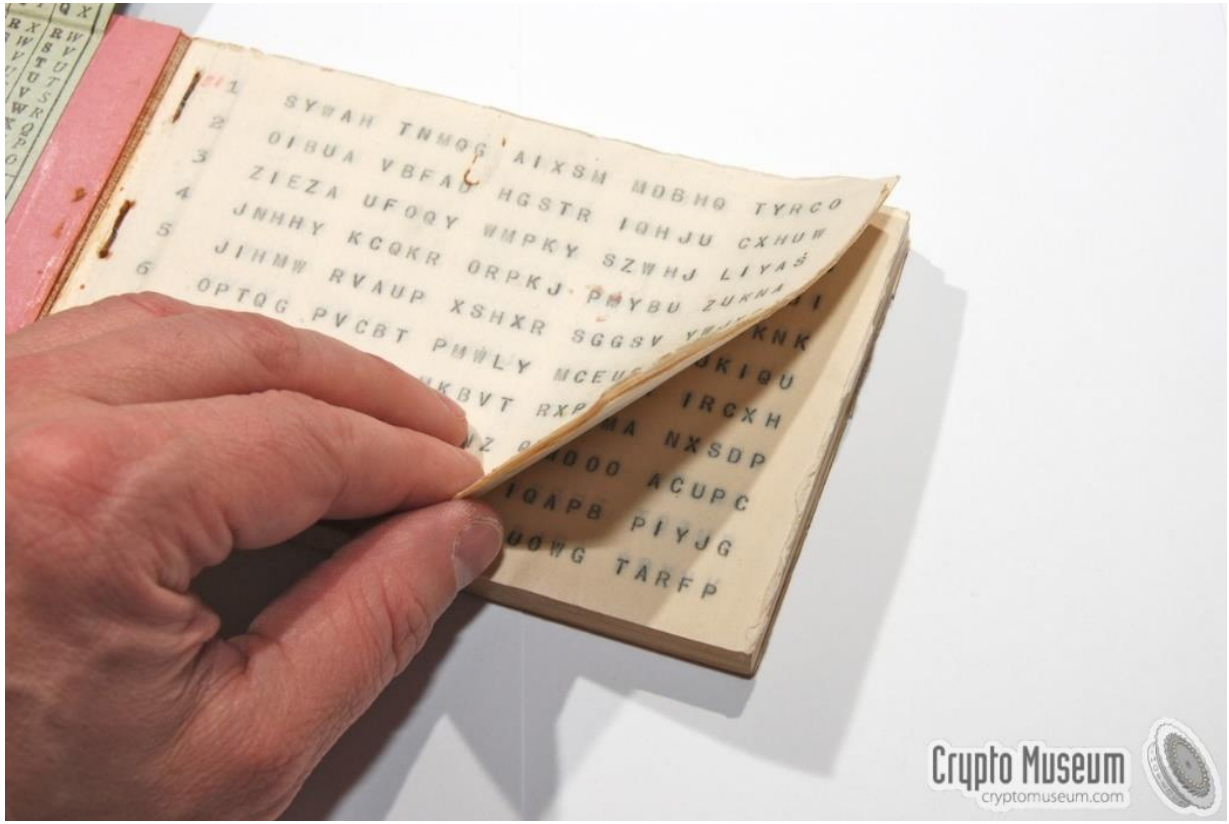
- Satu-satunya algoritma kriptografi sempurna aman (*perfect secrecy*) sehingga tidak dapat dipecahkan adalah *one-time pad (OTP).*

- OTP ditemukan pada tahun 1917 oleh Major Joseph Mauborgne.

- OTP mengatasi kelemahan pada *Vigenere Cipher. Vigenere Cipher* mengulang penggunaan kunci secara periodik → mudah ditemukan dengan metode Kasiski.
- Pada OTP, panjang kunci = panjang plainteks

Plainteks: `otpadalahcipheryangtidakbisadipecahkan`
Kunci: `trjkdndkdwerylgrgdkopcegyhbdwjbtrfhgvk`

- *One-time pad* (*pad* = kertas bloknot) berisi deretan huruf-huruf kunci yang dibangkitkan secara acak.



Sumber: https://www.cryptomuseum.com/crypto/otp/index.htm

CIHJT UUHML FRUGC ZIBGD BQPNI PDNJG LPLLP YJYXM
DCXAC JSJUK BIOYT MWQPX DLIRC BEXYK YKIMB TYIPE
UOLYQ OKOXH PIJKY DRDBC GEFZG UACKD RARCD HBYRI
DZJYO YKAIE LIUYW DFOHU IOHZY SRNDD KPSSO JMPQT
MHQHL OHQQD SMHNP HHOHQ GXRPJ XBXIP LLZAA YCMOG
AWSSZ YMFNI ATMON IXPBY FOZLE CYYSJ XZGPU CTFQY
HOYHU OCJGU QMWQY OIGOR BFHIZ TYFDB YBRMN XNLZC

- Pengirim dan penerima pesan memiliki salinan (*copy*) *pad* yang sama.

- Satu *pad* hanya digunakan sekali (*one-time*) saja untuk mengenkripsi pesan

  → itulah mengapa dinamakan *one-time pad*.

- Sekali *pad* telah digunakan, ia dihancurkan supaya tidak dipakai kembali untuk mengenkripsi pesan yang lain

  → menyulitkan kriptanalisis

Plainteks: `otpadalahcipheryangtidakbisadipecahkan`

Kunci: `trjkdndkdwerylgrgdkopcegyhbdwjbtrfhgvk`

- Aturan enkripsi dan dekripsi yang digunakan persis sama seperti pada *Vigenere Cipher*, bedanya tidak ada perulangan kunci secara periodik.

- Enkripsi: $c_i = (p_i + k_i) \bmod 26$

- Dekripsi: $p_i = (c_i - k_i) \bmod 26$

- **Contoh 1**:

  Plainteks: `onetimepad`

  Kunci: `tbfrgfarfm`

  Misalkan `A` = 0, `B` = 1, …, `Z` = 25.

  cipherteks: `HOJKOREGHP`

  yang dalam hal ini diperoleh sebagai berikut:

  (`o` + `T`) mod 26 = `H`

  (`n` + `B`) mod 26 = `O`

  (`e` + `F`) mod 26 = `J`, dst

- **Contoh 2:**

  Plainteks: `nantimalamsayatunggukamudidepanwarungkopi`

  Kunci: `gtrskncvbrwpoatqljfmxtrpjsrzolfhtbmaedpvy`

  Cipherteks: `TTELSZCGBDOPMAMKYPLGHTDJMAUDDLSDTSGNKNDKG`

- Kunci untuk OTP harus seluruhnya acak dan sepanjang pesan.

- Bagaimana jika kunci diambil dari teks yang panjang (misalnya tulisan di dalam novel, buku, berita, dan sebagainya)?

  - ini bukan lagi OTP (sebab tulisan di buku/novel/berita bukan acak)

  - tidak menghasilkan *perfect secrecy*

  - dapat dipecahkan

- Kunci di dalam OTP hanya dipakai sekali dan tidak pernah digunakan kembali. Bagaimana jika kunci dipakai untuk kedua kalinya?

  - ia bukan lagi *one-time pad*, tetapi *two-time pad*

  - tidak aman

- ***OTP* ini tidak dapat dipecahkan karena:**

  1. Kunci acak + plainteks yang tidak acak = cipherteks yang seluruhnya acak.

     Enkripsi: $c_i = (p_i + k_i) \bmod 26$
     Dekripsi: $p_i = (c_i - k_i) \bmod 26$

  2. Hanya terdapat satu kunci yang memetakan plainteks ke cipherteks, begitu juga sebaliknya.

- Mendekripsi cipherteks dengan beberapa kunci berbeda dapat menghasilkan plainteks yang bermakna, sehingga kriptanalis kesulitan menentukan plainteks mana yang benar.

- **Contoh 3**: Misalkan kriptanalis mencoba kunci `LMCCAWAAZD`
  untuk mendekripsi cipherteks `HOJKOREGHP`
  Plainteks yang dihasilkan: `SALMONEGGS`

  Bila ia mencoba kunci: `ZDVUZOEYEO`
  Plainteks yang dihasilkan: `GREENFIELD`

  Kriptanalis: ??????? (bingung sendiri ☺)

- Contoh ini menunjukkan bahwa untuk sembarang plainteks dan cipherteks hanya ada satu kunci yang memetakannya satu sama lain.

- Sebagai latihan, misalkan diberikan sebuah cipherteks:

  TLCYKUMGDFAWTZVOYKLENSZZHYZRW

  temukan kunci yang menghasilkan plainteks:

  mr johnson left his house last night

  lalu temukan kunci lain yang menghasilkan plainteks

  i saw the mysterious plane behind me

Kelemahan OTP

- Meskipun OTP menawarkan keamanan yang sempurna, tetapi ia tidak umum digunakan dalam aplikasi praktis (aplikasi komersil maupun aplikasi lainnya).

- Alasan:

  1. Tidak mangkus, karena panjang kunci = panjang pesan.
     Makin panjang pesan, makin besar ukuran kuncinya. Butuh komputasi yang berat untuk membangkitkan milyaran karakter-karakater yang benar-benar acak.

  2. Karena kunci dibangkitkan secara acak, maka ‘tidak mungkin’ pengirim dan penerima membangkitkan kunci yang sama secara bersamaan.

- *OTP* hanya dapat digunakan jika tersedia saluran komunikasi kedua yang cukup aman untuk mengirim kunci.

- Saluran kedua ini tidak boleh sama dengan saluran untuk mengirim pesan.

- Saluran kedua ini umumnya lambat dan mahal (misalnya lewat jalur darat, memakai kurir terpercaya dan tidak bisa dikenali).

Contoh Penggunaan OTP

- Perang dingin antara AS dan Uni Soviet (tahun 1940):
  - agen spionase Uni Soviet membawa kunci *one-time pad* ke AS
  - pesan-pesan rahasia dienkripsi dengan OTP dan dikirim dari AS
  - di Uni Soviet, kunci OTP yang sama digunakan untuk mendekripsi cipherteks

- *As a practical person, I've observed that one-time pads are theoretically unbreakable, but practically very weak. By contrast, conventional ciphers are theoretically breakable, but practically strong." -* ***Steve Bellovin***

Referensi utama :

>> Michael Felderer , Riccardo Scandariato (editor) - Exploring Security in Software Architecture and Design, 2018.

>> Nancy R. Mead, Carol Woody - Cyber Security Engineering_ A Practical Approach for Systems and Software Assurance-Addison-Wesley Professional (2016)

>> James Helfrich - Security for Software Engineers-CRC Press (2019)

>> Pete Loshin - Simple Steps to Data Encryption_ A Practical Guide to Secure Computing-Syngress (2013)

>> Tevfik Bultan,Fang Yu,Muath Alkhalaf,Abdulbaki Aydin (auth.) - String Analysis for Software Verification and Security (2017)


Any
Questions?