



1 of 17

BI-BEZ – Bezpečnost – Lab. cvičení 1

Tomáš Zahradnický, Jiří Buček

Katedra počítačových systémů, FIT ČVUT v Praze

- Úvod do software Mathematica (opakování)
- Substituční šifra
- Afinní šifry
- Transpoziční šifra



2 of 17

Velejemny uvod do software Mathematica

Ve cvicich bude vyuzivano software *Mathematica* pro demonstraci sifer, jejich kryptoanalyzy, atd., a proto je dulezite se se softwarem *Mathematica* naucit pracovat alespon na nejake zakladni bazi.

Mathematica je rozdelena na Front End (toto) a Kernel (neni videt). Front End posila prikazy oznacene In[cislo] do kernelu a vystup z kernelu je oznacen Out[cislo]. Prikaz In[cislo] spustite (poslete do kernelu k vyhodnoceni) pomoci klavesove kombinace SHIFT+ENTER, anebo ENTER na numericke klavesnici.

Pred tim, nez zacneme, *Mathematica* je case sensitive a velmi dba na typ zavorek a proto pozor. Kulate zavorky oznacuji prioritu vyhodnocovani, hranate urcuji argumenty funkce a slozene oznacuji vektory, matice a seznamy. Vice viz dale.

Nyni zkuste vyhodnoti nasledujici vyrazy:

```
In[1] := Mod[283, 17]
```

```
In[2]:= Sin[Pi / 2]

In[3]:= Sum[1 / i^2, {i, 1, ∞}]

In[4]:= Mod[102 * BEZInverse[102, 113], 113]
```



3 of 17

Inicializace worksheetu

Aby bylo možné použít programy v technických slajdech, je nutné provést inicializaci. Klepnete kamkoliv do bloku s programem a nechte ho vyhodnotit. Blok obsahuje definice funkcí, které budou použity pro šifrování, dešifrování a analýzu. Není nutné se těmito funkcemi zabývat.

```
In[35]:= << "BarCharts`";
NCharacters = 26;
SYMBOLS = Table[FromCharacterCode[65 + i], {i, 0, 57}];
ENGLISH = {0.0805726131341181993`1.9999999999999998,
  0.0125943599170806117`1.9999999999999998,
  0.0362576759103531896`1.99999999999999976,
  0.0380959831032189932`1.9999999999999998,
  0.1146790784996284273`1.9999999999999998,
  0.0240935581022411703`1.9999999999999998,
  0.0157233934368521923`1.9999999999999998,
  0.031212109359721516`1.9999999999999998,
  0.0741972073375836039`1.9999999999999998,
  0.001290726326905777`1.9999999999999998,
  0.0015254038408886455`1.9999999999999998,
  0.0429459850588649431`1.9999999999999998,
  0.0222943638283725115`1.9999999999999998,
  0.0746274494465521962`1.9999999999999998,
  0.0873782610396213869`1.9999999999999998,
  0.0389955802401533226`1.9999999999999998,
  0.0010169358939257637`1.99999999999999976,
  0.0751750303125122228`1.9999999999999998,
  0.0571830875738256346`1.9999999999999998,
  0.0912504400203387179`1.9999999999999998,
  0.0322290452536472797`1.9999999999999998,
  0.0081746000704032542`1.9999999999999998,
  0.0116165369421519928`1.99999999999999976,
  0.0017991942738686588`1.9999999999999998,
  0.0243282356162240388`1.9999999999999998,
  0.0007431454609457504`1.9999999999999998};
```

```

VelikostAbecedy[n_Integer] := Module[{}, NCharacters = n; Take[SYMBOLS, n]]
BEZRotChar[x_, amount_] := FromCharacterCode[
  Mod[(ToCharacterCode[x] - ToCharacterCode["A"]) + amount, NCharacters] +
  ToCharacterCode["A"]]
AXPBChar[x_, a_, b_] := FromCharacterCode[
  Mod[(ToCharacterCode[x] - ToCharacterCode["A"]) a + b, NCharacters] +
  ToCharacterCode["A"]]
AXPB[x_String, a_, b_] := StringJoin[
  (AXPBChar[#1, a, b] &) /@ Characters[BEZPrepText[x]]]
AXPBDecrypt[x_String, a_, b_] :=
  StringJoin[(AXPBCharDecrypt[#1, a, b] &) /@ Characters[x]]
BEZZnakPlusB[x_String, b_Integer] := (BEZRotChar[#1, b] &) /@ Characters[x]
AXPBCharDecrypt[x_, a_, b_] := FromCharacterCode[
  Mod[(ToCharacterCode[x] - ToCharacterCode["A"]) - b) BEZInverse[a, NCharacters],
  NCharacters] + ToCharacterCode["A"]]
BEZInverse[x_Integer, mod_] := Inverse[{x}, Modulus -> mod][1][1]
Caesar[x_String, posun_] := StringJoin[BEZZnakPlusB[x, posun]]
BEZPrepText[x_String] := FromCharacterCode[
  Select[{#1][1] &) /@ ToCharacterCode[Characters[ToUpperCase[x]]],
  0 <= #1 - ToCharacterCode["A"][1] <= NCharacters &]]
BEZAbsCetnosti[x_String] := Table[Length[
  Select[Characters[BEZPrepText[x]], #1 === SYMBOLS[[i] &]], {i, 1, NCharacters}]
BEZRelCetnosti[x_String] := N[ $\frac{\text{BEZAbsCetnosti}[x]}{\text{StringLength}[\text{BEZPrepText}[x]]}$ , 2]

RelCetnosti[x_String] :=
  TableForm[Module[{X}, X = BEZRelCetnosti[x]; Append[Table[Table[
    {SYMBOLS[[7 i + j]], X[[7 i + j]], {j, 1, 7}}, {i, 0, -1 + Floor[ $\frac{\text{NCharacters}}{7}$ ]}],
    Table[{SYMBOLS[[7 Floor[ $\frac{\text{NCharacters}}{7}$ ] + i]], X[[7 Floor[ $\frac{\text{NCharacters}}{7}$ ] + i]]},
    {i, 1, Mod[NCharacters, 7]}]]]]]
BEZGrafyRelCetnosti[x_String, y_String] := Module[{RC1, RC2},
  RC1 = BEZRelCetnosti[x]; RC2 = BEZRelCetnosti[y];
  BarChart[{RC1, RC2}, BarLabels -> SYMBOLS, BarEdges -> False,
  BarStyle -> {Directive[RGBColor[.1, .2, .7], Opacity[0.7]],
  Directive[RGBColor[0, .7, 0], Opacity[0.7]]},
  BarSpacing -> 0.1, Background -> GrayLevel[.9]]]
BEZGrafyRelCetnostiSAnglictinou[x_String] :=
  Module[{RC1, RC2}, RC1 = BEZRelCetnosti[x]; RC2 = ENGLISH;
  BarChart[{RC1, RC2}, BarLabels -> SYMBOLS, BarEdges -> False,
  BarStyle -> {Directive[RGBColor[.1, .2, .7], Opacity[0.7]],
  Directive[RGBColor[0, .7, 0], Opacity[0.7]]},
  BarSpacing -> 0.1, Background -> GrayLevel[.9]]]
RelCetnostiZBEZRelCetnosti[x_] :=
  TableForm[{Table[{FromCharacterCode[64 + i], x[[i]], {i, 7}},
  Table[{FromCharacterCode[64 + i], x[[i]], {i, 8, 14}},
  Table[{FromCharacterCode[64 + i], x[[i]], {i, 15, 21}},
  Table[{FromCharacterCode[64 + i], x[[i]], {i, 22, 26}}]}]
Pozice[x_String] := StringPosition[StringJoin[SYMBOLS], x][1][1] - 1
BEZPadString[x_String, n_Integer] := If[StringLength[x] < n,
  BEZPadString[StringJoin[x, "X"], n], If[StringLength[x] == n, x, Abort[]]]

```

```

BEZNumToPad[x_String, cols_Integer] :=
  Floor[(StringLength[x] + cols - 1) / cols] * cols
BEZGenerMatrix[x_String, cols_Integer] := Partition[Characters[
  BEZPadString[BEZPrepText[x], BEZNumToPad[BEZPrepText[x], cols]], cols]
Transpozice[x_String, cols_Integer] :=
  StringJoin[Flatten[Transpose[BEZGenerMatrix[BEZPrepText[x], cols]]]]

Print["Initialization done"]

```

4 of 17

Jednoduché sifry (Caesarova sifra)

Caesarova sifra je známa jako jednoduchá substitucní proudová (znaková) sifra, která provádí transformaci $y = |x + 3|_{26}$. Mejmě otevřený text (nechte vyhodnotit)

```
In[27]:= OT := "ANOPENTEXTTHATWILLGETTRANSFORMEDWITHCAESARCIPHER"
```

Zasifrovaný text bude:

```
In[28]:= ST = Caesar[OT, 3]
```

Text lze snadno desifrovat provedením opačné operace (odectením) posunu:

```
In[29]:= Caesar[ST, -3]
```

Kryptoanalýzu této sifry naleznete na další stránce.

5 of 17

Kryptoanalýza Caesarovy sifry

Relativní četnosti sifrovaného textu z minuleho slajdu jsou:

```
In[30]:= RelCetnosti[ST]
```

Relativní četnosti otevřeného textu z minuleho slajdu jsou:

```
In[31]:= RelCetnosti[OT]
```

6 of 17

Kryptoanalýza Caesarovy sifry (2)

Srovnáním relativních četností (červená pro sifrovaný text, modrá pro otevřený text) vidíme, že sifra způsobila posun relativních četností následujícím způsobem:

```
In[32]:= BEZGrafyRelCetnosti[ST, OT]
```

Z grafu vidíme, že pokud známe otevřený text, je velmi snadné uhodnout transformaci, která povede k desifrování textu. Stačí jen posunout četnosti tak, aby grafy splynuly. Pokud otevřený text k dispozici nemáme, musíme si vystačit například se vzorkem relativních četností pro jazyk, kterým předpokládáme, že je sifrovaný text psán. Předpokládáme, že sifrovaný

text je psan v anglictine, pro kterou máme relativní četnosti uložené v proměnné **ENGLISH**.



Kryptoanalýza Caesarovy šifry (3)

Relativní četnosti pro angličtinu jsou (získáno jako relativní četnosti z cca 20KB souboru):

```
In[33]:= RelCetnostizBEZRelCetnosti[ENGLISH]
```

Nyní můžeme udělat stejný krok, jako v minulém případě a srovnat relativní četnosti šifrovaného textu s relativními četnostmi pro angličtinu:

```
In[34]:= BEZGrafyRelCetnostiSAnglictinou[ST]
```



Úkol 1: Zjistete co se skrývá pod následujícím šifrovaným textem?

Neznámý šifrovaný text **ST** je:

```
In[64]:= ST = "BAGURNNGRRZCGAHZOREGUERRABGONQPBATENGHYNGVBAF"
```

```
In[280]:=
```

Provedte analýzu relativních četností a naleznete k němu otevřený text. Jde o šifru podobnou Caesarově šifře.

```
In[65]:= BEZGrafyRelCetnostiSAnglictinou[ST]
```

Navod: Srovnajte relativní četnost. Proměnná **ST** je globální, takže můžete použít pomůcky uvedené na předchozích slajdech.



Šifry typu $|ap + b|_m$

Caesarova šifra je speciálním případem afinní šifry $|ap + b|_m$ a je definována jako $|p + b|_m$. Šifra tedy představuje substituci, avšak nyní již tato substituce neposunuje graf relativních četností, ale dochází k důkladnějšímu "promíchání" (viz graf relativních četností). Celou věc si ukážeme na příkladu:

```
In[66]:= OT = "THISISYETANOTHERCIPHERTEXTTHATWEWILLENCIPHER"
```

```
In[67]:= NCharacters = 29
          ST = AXPB[OT, 6, 19]
```

```
In[69]:= BEZGrafyRelCetnosti[ST, OT]
```

Šifrovaný text můžeme dešifrovat obrácením šifrovacího předpisu a to je-li $y = |ap + b|_m$ bude $p = |a^{-1}(y - b)|_m$.

```
In[70]:= AXPB[ST, 5, 21]
```

Poznámka: Příkazem **NCharacters=29** se rozšířila abeceda na 29 symbolů; jejich seznam viz dole.

Všimněte si, že jsme před použitím transformace $|ap + b|_m$ rozšířili vstupní abecedu na 29 znaků z 26 použitím příkazu **NCharacters=26**. Proč byla tato operace nutná a co se stane, když **NCharacters** zustane 26?

Promenna **SYMBOLS** obsahuje celou abecedu, která může být rozšířena až na 58 znaků. Pro vypsání abecedy použijte pro šifrování můžeme použít funkci softwaru *Mathematica* **Take**, která vezme prvních N znaků ze seznamu **SYMBOLS**, kde $N=N\text{Characters}$.

```
In[71]:= Take[SYMBOLS, NCharacters]
```



Ukol 2: Kryptoanalýza šifry $|ap + b|_m$

1. Vyberte náhodné konstanty a a b a pro $N\text{Characters}=29$ s nimi anglický text alespoň o 100 znacích. Takový text můžete najít například v libovolné dokumentaci k softwaru. Az budete mít šifrový text, poskytnete ho sousedovi (např. e-mailem) ale nesdělujte mu hodnoty konstant!

```
In[355]:=
NCharacters = 29;
OT = "NOTICE: This software will not perform or complete any actual financial
      transactions. You must obtain a separate commercial use license
      from MindVision to use eSellerate to conduct electronic commerce.
      Even though it appears to be fully functional, it is not fully
      functional, IT WILL NOT CONDUCT FINANCIAL TRANSACTIONS, and the
      software provided under this Agreement is NOT FOR DISTRIBUTION. Under
      no circumstances shall MindVision be liable for any transactions
      utilizing the Software under this Evaluation License Agreement.";
a = 9;
b = 23;
```

```
In[359]:=
ST = AXPB[OT, a, b]
```



Ukol 2: Kryptoanalýza šifry $|ap + b|_m$ (2)

2. Az obdržíte šifrový text od souseda, přiřadte jeho hodnotu do proměnné ST:

```
In[360]:=
ST =
"YEUIMBU] ILLEKUSXCBSIGGYEUNBCKECPCEPNGBUBXYHXMUAXGKIYXYMIXGUCXYLXMUIEYLHEAPAL\
UEDUXIYXLBNXCXUBMEPPBCMIXGALBGIMBYLBKCEPPIYVJILIEYUEALBBLBGGBCXUBUEMEYVAM\
UBGBMUCEYIMMEPPBCMBBJBYU] EAT] IUXNNBXCLUEDBKAGGHKAYMUIEYXGIUILEYUKAGGHKAYMU\
IEYXGIUSIGGYEUMEYVAMUKIYXYMIXGUCXYLXMUIEYLXYVU] BLEKUSXCBNCEJIVBVAYVBCU]
ILXTCBBPBYUILEYUKECVILUCIDAUIEYAYVBCYEMICMAPLUXYMBLL]
XGGPIYVJILIEYDBGIXDGBKECXHUCXYLXMUIEYLAUIGIQIYTU] BLEKUSXCBAYVBCU]
ILBJXGAXUIEYGIMBYLBXTCBPBYU";
```

Nyní provedte analýzu četnosti:

```
In[361]:=
RelCetnosti[ST]
```

A srovnejte ji s cetnostmi pro anglictinu:

```
In[362]:=
RelCetnostiZBEZRelCetnosti[ENGLISH]
```

12 of 17

Ukol 2: Kryptoanalýza sifry $|ap + b|_m$ (3)

Z analýzy relativních četností můžeme zjistit, že nejčastější písmena pro angličtinu jsou T a E. Toho můžeme použít pro zjištění desifrovacího klíče. Vybereme tedy 2 nejčastější písmena z analýzy šifrovaného textu a zkusíme je namapovat na T a E. Řešením soustavy 2 rovnic o 2 neznámých vypočteme neznámé koeficienty a a b .

Rekneleme ze pro nás příklad vidíme, že nejčastější jsou písmena U a B. Zkusíme tedy předpokládat, že:

$$U = |aT + b|_{29}$$

$$B = |aE + b|_{29}$$

Tedy, z T neznámou transformací vznikne U a z E toutéž transformací vznikne B . Abychom určili a a b , musíme už jen vyřešit tyto dvě rovnice například dosazovací metodou:

$$b = |U - aT|_{29}$$

$$B = |aE + |U - aT|_{29}|_{29}$$

Protože nezáleží, kdy redukci mod 29 provedeme, můžeme vztah přepsat jako:

$$B = |a(E - T) + U|_{29}$$

$$a = |(B - U) * (E - T)^{-1}|_{29}$$

$$b = |U - (B - U) * (E - T)^{-1} T|_{29}$$

Nyní můžeme předpis zkusit (pozn. pokud budete počítat na papíře, nezapomente že A odpovídá 0, B 1, atd.):

13 of 17

Ukol 2: Kryptoanalýza sifry $|ap + b|_m$ (4)

Pro jednoduchost jsou zde rovnice implementovány do systému *Mathematica* a vypočtené konstanty se uloží do proměnných c a d .

```
In[61]:= c = Mod[(Pozice["B"] - Pozice["U"]) *
    BEZInverse[Pozice["E"] - Pozice["T"], NCharacters], NCharacters]
d = Mod[Pozice["U"] - c * Pozice["T"], NCharacters]
```

Nyní se můžeme pokusit text desifrovat:

```
In[365]:=
AXPBDDecrypt[ST, c, d]
```

14 of 17

Transpozice

Substituce způsobuje konfuzi nahradou jednoho znaku znakem jiným a sifry založené na ní jsou zranitelné frekvenční analýzou. Proto obvykle substituci kombinujeme s transpozicí, která preskupuje pořadí písmen v textu (difuze). To se v praxi provádí zapsáním otevřeného textu do matice po řádcích a přečtení po sloupcích. Ukážme si to na příkladě textu "PLEASE SEND MONEY", který nejprve doplníme výplní (X) na délku, která obsadí celou matici. Tak získáme matici:

$$\begin{pmatrix} P & L & E & A & S & E \\ S & E & N & D & M & O \\ N & E & Y & X & X & X \end{pmatrix}, \text{ kterou transponujeme } \begin{pmatrix} P & S & N \\ L & E & E \\ E & N & Y \\ A & D & X \\ S & M & X \\ E & O & X \end{pmatrix} \text{ a přečteme text po řádcích, čímž dostáváme:}$$

PSNLEEEENYADXSMXEOX. V kombinaci se substitucí například pomocí afinní sifry je výsledná šifra posílena. Transpozici si můžete vyzkoušet voláním funkce `Transpozice[retezec, pocet sloupcu]`:

```
In[63]:= NCharacters = 26;
ST = Transpozice["THE GOLD IS BURIED IN ORONO", 6]
```

Pokud bychom chtěli vidět matici, můžete použít funkci `BEZGenerMatrix[retez, pocet sloupcu]`. Tím zároveň uvidíme množství výplně, které se přidalo na zarovnání na potřebnou délku.

```
In[65]:= BEZGenerMatrix["THE GOLD IS BURIED IN ORONO", 6] // MatrixForm
```



Transpozice (2)

Detranspozice se provádí obrácením předpisu, tedy zadáním počtu řádků výsledné matice:

```
In[66]:= OT = Transpozice[ST, 4]
```

Je zřejmé, že transpozice **NEMA** vliv na frekvenční uspořádání a graf otevřeného i šifrovaného textu bude pro analýzu relativní četnosti naprosto identický. To si můžeme ukázat v následujícím příkladu:

```
In[67]:= OT
ST
```

```
In[69]:= BEZGrafyRelCetnosti[OT, ST]
```



Kryptoanalýza transpozice

Kryptoanalýzu transpoziční sifry provádíme na základě bigramové analýzy. V šifrovaném textu hledáme bigramy avšak mezi jednotlivými znaky bigramu bývá zpravidla několik (někdy mnoho) dalších znaků. Pro angličtinu je typické hledat bigramy s největší četností, což jsou: TH, HE, AN, RE, ER, IN, ON, AT, ND, ST, ES, EN, OF, TE a ze vzdálenosti znaku bigramu se snažíme stanovit parametry transpozice, které byly použity. Pro šifrovaný text **TDIRHIEOESDNG-BIOOUNXLROX** vidíme ihned bigramy TH a HE z čehož usoudíme, že měla matice 4 řádky:

```
In[70]:= Transpozice["TDIRHIEOESDNG-BIOOUNXLROX", 4]
```

Další možností je faktorizovat délku textu a tím zjistit všechny delitele délky textu a postupně je zkusit.


```
In[71]:= FactorInteger[StringLength["TDIRHIEOESDNGBIOOUNXLROX"]]/.
      {x_Integer, y_Integer} → HoldForm[x^y]
```

Zde tento případ vidíme, že délka textu je 24 znaků. Z toho můžeme usoudit, že pro transpozici mohla být: 2×12 , 3×8 , 4×6 , 6×4 , 8×3 , 12×2 . Jiné kombinace neexistují. Pro náš případ byla použita transpozice 6×4 .

Pokud text nebyl tak dlouhý, jak byl zadán předpis, musel být doplněn výplní (padding). Ze znalosti funkce, která transpozici provádí, víme, že tato funkce doplňuje na konec textu písmena X, dokud nezarovná text na potřebnou délku a to je skutečnost, kterou můžeme využít k prolomení transpozice, protože vzdálenost této výplně (zvláště v případě, že bylo doplněno více znaků X) udává transpozici konstantu, která byla použita při šifrování. Délka textu/tato konstanta udává detranspozici konstantu.

Pro náš text vidíme, že vzdálenost paddingu X je 4 znaky, z čehož rovnou plyne konstanta pro detranspozici.

Upozornění: Pokud byla transpozice použita vícenásobně, je její kryptoanalýza značně obtížnější!!

```
In[72]:= Transpozice[Transpozice["THEGOLDISBURIEDINORONO", 4], 3]
```



Ukol 3: Kryptoanalýza transpozice

Napište kus anglického textu o délce 30-50 znaků a náhodně zvolte transpozici konstantu. Proveďte nad tímto textem transpozici s vami zvolenou konstantou a výsledný šifrovaný text pošlete sousedovi.

```
In[73]:= OT = "DEFEATTHETRANSPOSITIONCHALLENGE"
```

```
In[74]:= Transpozice[OT, 9]
```

Až obdržíte text od souseda, přiřadte ho to této proměnné:

```
In[75]:= ST = "DTTEERINFAOGENNEASCXTPHXTOAXHSLXEILX"
```

Nyní se snažte zjistit, jakou transpozici váš soused použil. Využijte metod popsanych na minulem slajdu. Pro jednoduchost faktORIZACI opakujeme nyní pro proměnnou šifrovaného textu:

```
In[76]:= FactorInteger[StringLength[ST]] /. {x_Integer, y_Integer} → HoldForm[x^y]
```

```
In[77]:= Transpozice[ST, 4]
```