Kryptoanaliza w .rb

Na poczatku był cyphertext...

cnVieSBqZXN0IGZham55IQ==

Podobny do base64...

[plain] => [base64]

'r' => 'cg=='

'ru' => 'cnU='

'rub' => 'cnVi'

'ruby' => 'cnVieQ=='

```ruby
b64_regexp = /^([A-Za-z0-9=]){4,}$/
if b64_regexp =~ "cnVieSBqZXN0IGZham55IQ=="
  p "Works!"
end

# => "Works!"
```

```ruby
require 'base64'
Base64.decode64 "cnVieSBqZXN0IGZham55IQ=="

# => "ruby jest fajny!"
```

ZWhvbCB3cmZnIHNud2FseiB3cm1seHZyeiBjZWJ0Z
W56YmpuYXZuIHFiIHVucHhiam5hdm4gdiBtbm9uamw=

```ruby
require 'base64'
cypher = [z-poprzedniego-slajdu]
if cypher =~ /^([A-Za-z0-9=]){4,}$/
  p Base64.decode64 cypher
end

# => "ehol wrfg snwalz wrmlxvrz
#     cebtenzbjnavn qb unpxbjnavn vmnonjl"
# WTF?
```

Wyglada jak szyfr podstawieniowy (jak cezara)

Analiza jezyka

Czyli częstość występowania liter w słowniku

```ruby
def statistic(file)

result = {}
result.default = 0

file = File.new(file, "r")
file.each_line do |line|
  line.chomp!
  line.each_char do |chr|
    result[chr] += 1
  end
end
```

Statytyka języka polskiego

{"a"=>2695234, "b"=>623287, "c"=>1239361, "i"=>2397411, "d"=>677679, "ń"=>46124, "s"=>991022, "k"=>953930, "e"=>2006028, "g"=>404793, "m"=>837763, "n"=>2068393, "o"=>2370256, "w"=>1375648, "ó"=>107526, "z"=>1425389, "y"=>1189959, "u"=>680712, "l"=>729218, "t"=>835145, "j"=>475555, "ż"=>156373, "r"=>1356066, "h"=>185591, "p"=>922455, "ś"=>215564, "ł"=>695373, "ą"=>235185, "f"=>135852, "v"=>212, "ć"=>7951, "ę"=>158667, "ź"=>21221, "q"=>8, "x"=>16}

Kolejność liter w języku polskim

["a", "i", "o", "n", "e", "z", "w", "r", "c", "y", "s", "k", "p", "m", "t", "l", "ł", "u", "d", "b", "j", "g", "ą", "ś", "h", "ę", "ż", "f", "ó", "ń", "ź", "ć", "v", "x", "q"]

Statystyka cypher'a

{"e"=>3, "h"=>1, "o"=>2, "l"=>3, " "=>8, "w"=>3, "r"=>3, "f"=>1, "g"=>1, "s"=>1, "n"=>9, "a"=>3, "z"=>3, "m"=>2, "x"=>2, "v"=>4, "c"=>1, "b"=>4, "t"=>1, "j"=>3, "q"=>1, "u"=>1, "p"=>1}

Kolejność cyphera

["n", " ", "v", "b", "w", "l", "r", "j", "z", "e", "a", "o", "x", "m", "s", "f", "c", "g", "t", "h", "q", "u", "p"]

```
swap = {}
swap.default = "[?]"

cypher_stat.each_with_index do |item, index|
  swap[dic_stat[index]] = item
end
```

Odzyskane podstawienie ;-)

{"a"=>"m", "i"=>" ", "o"=>"s", "n"=>"g", "e"=>"j", "z"=>"r", "w"=>"u", "r"=>"c", "c"=>"e", "y"=>"a", "s"=>"h", "k"=>"p", "p"=>"t", "m"=>"x", "t"=>"b", "l"=>"z", "ł"=>"l", "u"=>"n", "d"=>"q", "b"=>"o", "j"=>"v", "g"=>"f", "ą"=>"w"}

Wynik!

h[?]zs[?]au[?]q[?]mcajsf[?]aurs[?][?]uf[?]whxehcfxbcj[?]c[?][?]x[?]ocv[?]xbcj[?]c[?][?][?]rczcbs

LOL

```ruby
require 'rot13'
cypher.rot13

# => ruby jest fajnym jezykiem
#  programowania do hackowania i zabawy
```

Jesteś administratorem.

Za pomocą RSA zakodowano kod PIN maszyny do kawy i przesłano do pracowników.

Każdy pracownik ma swój klucz publiczny.

Ty nie dostałeś swojego PINu -- co robić? (maszyna blokuje sie po 3 niepoprawnych kodach)

Przechwytujesz szyfrogramy

I oczywiście kryptoanaliza!

```ruby
# public key
pk = { :n => 3233, :e => 17 }
# private key
prk = { :n => 3233, :d => 2753 }

# pin-code
pin = 2563
cypher = 2563**(pk[:e])%pk[:n]
# => 2258
plain = cypher**(prk[:d])%prk[:n]
# => 2563
```

```ruby
pk = { :n => 3233, :e => 17 }
# cypher == 2258

9999.times do |pin|
  puts pin if pin**(pk[:e])%pk[:n] == 2258
end
```

```ruby
pk = { :n => 3233, :e => 17 }
# cypher == 2258

9999.times do |pin|
  puts pin if pin**(pk[:e])%pk[:n] == 2258
end

# => 2563, 5796, 9029
# MUHAHAHAHAHA! darmowa kawa ;p
```

Pomysł

Stworzenie prostego i używalnego narzędzia do kryptoanalizy w ruby na kolejnym hackatonie

7 maj

Pytania?

Kryptoanaliza w .rb

Na poczatku był cyphertext...

cnVieSBqZXN0IGZham55IQ==

Podobny do base64...

'r' => 'cg=='

'ru' => 'cnU='

'rub' => 'cnVi'

'ruby' => 'cnVieQ=='

```ruby
b64_regexp = /^([A-Za-z0-9=]){4,}$/
if b64_regexp =~ "cnVieSBqZXN0IGZham55IQ=="
  p "Works!"
end

# => "Works!"
```

```ruby
require 'base64'
Base64.decode64 "cnVieSBqZXN0IGZham55IQ=="

# => "ruby jest fajny!"
```

ZWhvbCB3cmZnIHNud2FseiB3cm1seHZyeiBjZWJ0Z
W56YmpuYXZuIHFiIHVu\ncHhiam5hdm4gdiBtbm9uamw=

```ruby
require 'base64'
cypher = [z-poprzedniego-slajdu]
if cypher =~ /^([A-Za-z0-9=]){4,}$/
  p Base64.decode64 cypher
end

# => "ehol wrfg snwalz wrmlxvrz
#     cebtenzbjnavn qb unpxbjnavn vmnonjl"
```

```ruby
def statistic(file)

result = {}
result.default = 0

file = File.new(file, "r")
file.each_line do |line|
  line.chomp!
  (line.size - 1).times do |time|
    result[line[time].chr] += 1
  end
end
```

Statytyka języka polskiego

{"a"=>2695234, "b"=>623287, "c"=>1239361, "i"=>2397411, "d"=>677679, "ń"=>46124, "s"=>991022, "k"=>953930, "e"=>2006028, "g"=>404793, "m"=>837763, "n"=>2068393, "o"=>2370256, "w"=>1375648, "ó"=>107526, "z"=>1425389, "y"=>1189959, "u"=>680712, "l"=>729218, "t"=>835145, "j"=>475555, "ż"=>156373, "r"=>1356066, "h"=>185591, "p"=>922455, "ś"=>215564, "ł"=>695373, "ą"=>235185, "f"=>135852, "v"=>212, "ć"=>7951, "ę"=>158667, "ź"=>21221, "q"=>8, "x"=>16}

Kolejność liter w języku polskim

["a", "i", "o", "n", "e", "z", "w", "r", "c", "y", "s", "k", "p", "m", "t", "l", "ł", "u", "d", "b", "j", "g", "ą", "ś", "h", "ę", "ż", "f", "ó", "ń", "ź", "ć", "v", "x", "q"]

Statystyka cypher'a

{"e"=>3, "h"=>1, "o"=>2, "l"=>3, " "=>8, "w"=>3, "r"=>3, "f"=>1, "g"=>1, "s"=>1, "n"=>9, "a"=>3, "z"=>3, "m"=>2, "x"=>2, "v"=>4, "c"=>1, "b"=>4, "t"=>1, "j"=>3, "q"=>1, "u"=>1, "p"=>1}

Kolejność cyphera

["n", " ", "v", "b", "w", "l", "r", "j", "z", "e", "a", "o", "x", "m", "s", "f", "c", "g", "t", "h", "q", "u", "p"]

Odzyskane podstawienie ;-)

{"a"=>"m", "i"=>" ", "o"=>"s", "n"=>"g", "e"=>"j", "z"=>"r", "w"=>"u", "r"=>"c", "c"=>"e", "y"=>"a", "s"=>"h", "k"=>"p", "p"=>"t", "m"=>"x", "t"=>"b", "l"=>"z", "ł"=>"l", "u"=>"n", "d"=>"q", "b"=>"o", "j"=>"v", "g"=>"f", "ą"=>"w"}

Wynik!

h[?]zs[?]au[?]q[?]mcajsf[?]aurs[?][?]uf[?]whxehcfxbcj[?]c[?][?]x[?]ocv[?]xbcj[?]c[?][?][?]rczcbs

LOL

```ruby
require 'rot13'
cypher.rot13

# => ruby jest fajnym jezykiem
#  programowania do hackowania i zabawy
```

Jesteś administratorem. Za pomocą RSA zakodowano kod PIN do maszyny do kawy (każdy pracownik ma swój klucz publiczny). Ty go nie dostałeś -- co robić?

Przechwytujesz szyfrogramy

I oczywiście kryptoanaliza!