Niebezpieczna Kryptografia - zadania kwalifikacyjne

Michał Radwański

Oto zadania kwalifikacyjne. Pierwsze z nich jest do rozwiązania "offline", należy podzielić się ze mną odpowiedzią. Dwa pozostałe wymagają interakcji z siecią. Rozwiązania (czyli odpowiedź oraz opis metody wraz z ewentualnym skryptem automatyzującym atak) powinny być wysłane na mój adres. Każde z zadań jest ocenianie na 10 punktów. Jeśli nie umiesz zrobić zadania w całości, nie przejmuj się - przy ocenianiu będę zwracał uwagę na poczynione obserwacje. Jeśli nie wiesz jak się zabrać za zadanie, albo gdzieś utknąłeś(łaś) – pisz – będę udzielać pomocy, gdy zobaczę postępy.

Proszę wysłać rozwiązania na adres mailowy `lisklusownik+www15@gmail.com`

Kody źródłowe związanie z każdym zadaniem są także dostępne tutaj: https://gist.github.com/enedil/dd8d0c08a014f52584c7bb700d379f80

Zadanie 1

Poniższym programem zaszyfrowano wiadomość. Musisz ją odzyskać.

```
#!/usr/bin/env python3
import random
from gmpy2 import is_prime, invert
# Jeśli dostajesz błąd "ImportError: No module named gmpy2", możesz to naprawić
# dzięki poleceniu
# python3 -m pip install gmpy2

from sekrety import wiadomosc
# tego z kolei nie ma, tutaj zapisałem sekretną wiadomość

pt = int.from_bytes(wiadomosc.encode(), 'big')
size = 1024

p = random.randint(2**(size - 1), 2**size)
while not is_prime(p):
    p += 1
q = p * 17 + random.randint(0, 2**(size//2 + 15))
while not is_prime(q):
    q += 1

N = p*q
e = 0x10001
d = invert(e, N - p - q + 1) # klucz prywatny
ct = pow(pt, e, N)

print('Klucz publiczny (N, e) =', (N, e))
print('Szyfrogram ct =', ct)
```

Wynik działania:

```
Klucz publiczny (N, e) = (46198647445938471459242186654372170980337503444513807821850003
```

```
19510151129239279744639523280538212649695913261714013519197951931548284533494500024010753801257003272101850146854355088075700441118776196620707125013523668445591877515971814651600547544972610977803839460185676002043108287797472816963533980504031837155351685792437853595867603139269714975258993883204742508767990834568828969733874353997493877356209220053348989942879066845336474391239713005061909934390788978355910571700546136227655575681282983550382419594258191174238245246649704999999135272617243942149894604105972212652020093968344746616388185935920408751, 65537)
Szyfrogram ct = 43833292712137503143707806374902808582555770336122195003018785902554227599869438170993986225904039816517966422586652855508241829078230516362307079670526802486292846641995841148789887425408110986909140180925794746409301306463086265355517123554571000090206972160421246577396422319932119265435245018699366776553060907745177922203246890314116108222430979157390387875736569672272591597049791056988095570266514012773730726681215214808761638034974925739377541528621976623471717388858462734250259978908286590658081321857514880948028960010411535667937980537593990247114602125901781542073248025120596806462740422879235477428625
```

Zadanie 2

Odsyskaj sekretną wiadomość. Oto kod serwera, który jest uruchomiony pod adresem http://students.mimuw.edu.pl/~mr395415/www15/insecure_crypto/zadanie2.cgi

```python
#!/usr/bin/env python3
import cgi
import os
import json
from base64 import b64decode, b64encode
from Crypto.Cipher import AES
from Crypto.Util.strxor import strxor
from Crypto.Util.Padding import pad, unpad
from Crypto.Util.number import long_to_bytes

from secret import key, secret_message

block_length = 16
aes = AES.new(key, AES.MODE_ECB)

def chunk(data, length):
    return [data[i:i+length] for i in range(0, len(data), length)]

def sign(msg, iv=None):
    msg = pad(msg, block_length)
    if not iv:
        iv = os.urandom(block_length)
    blocks = chunk(msg, block_length)

    signature = iv
    for block in [long_to_bytes(len(blocks), block_length)] + blocks:
        signature = aes.encrypt(strxor(signature, block))
    return iv, signature

def verify(msg, iv, signature):
    _, computed_signature = sign(msg, iv)
    return computed_signature == signature

msg_ = b'{"authenticated":false,"text":"oto_przyklad_uzycia_podpisu"}'
```

```
iv_, sig_ = sign(msg_)

print("Content-type: text/html")
print()
print( """
<html>
<head><title>Zadanie 2: AES CBC MAC</title></head>
<meta charset="utf-8">
<body>
""")
print("""
<p> sign(%r) == b64decode(%r), b64decode(%r) </p>
""" % (msg_, b64encode(iv_), b64encode(sig_)))

form = cgi.FieldStorage()
message = form.getvalue('message', '')
iv = form.getvalue('iv', '')
signature = form.getvalue('signature', '')

print('<p>')
def x(message, iv, signature):
    try:
        message = b64decode(message)
        iv = b64decode(iv)
        signature = b64decode(signature)
    except Exception as e:
        print('niepoprawne kodowanie: ', e)
    if len(iv) != 16 or len(signature) != 16:
        print('Zła długość.')
        return

    try:
        if verify(message, iv, signature):
            d = json.loads(message)
            if d['authenticated']:
                print('Oto sekretna wiadomość: ', secret_message)
            else:
                print('Tym razem niestety nic')
        else:
            print('Niepoprawny podpis.')
    except Exception as e:
        print(e)
if message and iv and signature:
    x(message, iv, signature)

print('</p>')

print("""


  <p>Wszystkie rzeczy trzeba najpierw zakodować za pomocą base64!

  <form method="post">
    <p>message: <input type="text" name="message"/></p>
    <p>iv: <input type="text" name="iv"/></p>
    <p>signature: <input type="text" name="signature"/></p>
```

```
  <input type="submit">
  </form>

</body>

</html>
""")
```

Zadanie 3

Odzyskaj ukrytą liczbę (a może znaczy ona także coś więcej?), skrywaną przez ten kod. Serwer jest uruchomiony pod adresem http://students.mimuw.edu.pl/~mr395415/www15/insecure_crypto/zadanie3.cgi

```
#!/usr/bin/env python3
import cgi

from secret import N, e, d

def encrypt(m):
    return pow(m, e, N)

def decrypt(c):
    return pow(c, d, N)

print("Content-type: text/html\n")
print( """
<html>
<head><title>Zadanie 3: RSA</title></head>
<meta charset="utf-8">
<body>
""")
print("""
<p>
Oto klucz publiczny: <br/>
N = %d <br/>
e = %d <br/>
</p>
""" % (N, e))
print("""
<p>
Kluczem tym zaszyfrowano pewną wiadomość (jest to liczba): <br/>
92806673530033597076124068735316514447716393066037
75902153871511726570230880920145069356785548452115408131 56
</p>
""")

form = cgi.FieldStorage()
szyfrogram = form.getvalue('szyfrogram', '')

if szyfrogram:
    try:
        dec = decrypt(int(szyfrogram))
        print("Po odszyfrowaniu, stwierdzam że odpowiadająca jemu odszyfrowana liczba jest ", end='')
        if dec % 2:
            print('parzysta.')
```

```
        else:
            print('nieparzysta.')
    except Exception as e:
        print("Wystąpił błąd:", e)

print(
 """<form method="post">
    <p>Wprowadź szyfrogram: <input type="text" name="szyfrogram"/></p>
  <input type="submit">
  </form>

</body>

</html>""")
```