d-phi-enc (HackTM CTF)

Кулигин Данил и Борзенко Михаил

БФУ им. Иммануила Канта ИФМНиИТ
Компьютерная безопасность, 3 курс

1 июля 2023 г.

Условие задачи d-phi-enc

Описание с CryptoHack

- Описание к заданию: *"В CTF есть много людей, которые ошибочно шифруют p, q в RSA. Но на этот раз..."*

- Так же к задаче прилагаются файлы *chall.py* и *output.txt*.output.txt содержит `n` - произведение p и q, `enc_d` - зашифрованная секретная экспонента, `enc_phi` - зашифрованная функция Эйлера, `enc_flag` - зашифрованное сообщение. В *chall.py* содержится код для генерации этих значений и открытая экспонента e

- Понятно, что нам нужно каким-то образом извлечь `d` или `phi` на основе `enc_d` и `enc_phi`.

- Для начала проведем анализ данных нам чисел и проверим криптосистему на уязвимость к изученным нами атакам, в числе которых: Атака повторным шифрованием, Атака Винера, Атака встреча посередине, Атака методом Ферма.

  Нам даны:
  `n` = 2447638356...
  `enc_d` = 2385197103...
  `enc_phi` = 3988439673...
  `enc_flag` = 2403368891...
  `e` = 3
- Упрощая, нам дано ***n*** длиной 2048 бит. Также мы знаем, что длина ***p*** и ***q*** равна 1024 бита. Очевидно, что разложить ***n*** на множители вряд ли будет эффективным решением.

- Атака “встреча посередине” не будет успешной, т. к. обычно она срабатывает при малой длине сообщения ($l < 64$ бит) и наличии разложения на два примерно равных множителя, битовая длина которых меньше $l/2$.

- Атака методом Ферма также не принесёт результатов, т. к. данная атака могла бы быть успешна, если p и q “близкие” друг к другу чилса (половина старших цифр числа равны).

- Атака Винера гарантированно успешна, при $d < d_{\text{кр}}$, где $d_{\text{кр}} = n^{1/4}/\sqrt{2a}$, $a = (h+1)/\sqrt{h}$, $h = p/q$.

- Как мы можем заметить, для вычисления $d_{\text{кр}}$ требуется знать $\boldsymbol{p}$ и $\boldsymbol{q}$, а мы их пока что не знаем, но мы знаем точно, что попытка атаки Винера (метод которой заключается в разложении дроби $\boldsymbol{e}/\boldsymbol{n}$ в цепную дробь для последующего нахождения $\boldsymbol{d}$) не принесла результата.

- Атака повторным шифрованием также не дала быстрого результата, из чего можем сделать вывод, что порядок $\boldsymbol{e}$ по модулю сообщения достаточно велик.

Решение задачи d-phi-enc

Теоретические рассчёты

- Из алгоритма работы RSA мы знаем, что ***d*** это обратное число к ***e*** по модулю $\varphi(n)$.

$$e \cdot d1 \equiv \pmod{\varphi(n)}$$
$$e \cdot d = k1 \cdot \varphi(n) + 1$$

- Т. к. ***d*** $<$ ***n*** и ***e*** $= 3$ можем сделать вывод, что $0 < k1 < 3$. Далее распишем d_{enc} используя сравнение выше.

$$d_{enc} \equiv d^3 \pmod{n}$$
$$e^3 \cdot d_{enc} \equiv e^3 \cdot d^3 \pmod{n}$$
$$27 d_{enc} \equiv (e \cdot d)^3 \pmod{n}$$
$$27 d_{enc} \equiv (k1 \cdot \varphi(n) + 1) \pmod{n}$$
$$27 d_{enc} \equiv (k1^3 \cdot \varphi^3(n) + 3k1^2 \cdot \varphi^2(n) + 3k1 \cdot \varphi(n) + 1) \pmod{n}$$

Решение задачи d-phi-enc

Теоретические рассчёты

- Заменим $\varphi^3(n)$ на φ_{enc}, т. к.$\boldsymbol{e} = 3$, а сравнение выполняется по модулю $\boldsymbol{n}$, также как и шифрование.

$$27d_{enc} \equiv (k1^3 \cdot \varphi_{enc} + 3k1^2 \cdot \varphi^2(n) + 3k1 \cdot \varphi(n) + 1) \pmod{n}$$

- Перенеся всё в одну сторону можно получить квадратное сравнение от $\varphi(n)$

$$3k1^2 \cdot 2(n) + 3k1 \cdot (n) + k1^3 \cdot \varphi_{enc} - 27d_{enc} + 1 \equiv 0 \pmod{n}$$

- Все переменные кроме $\varphi(n)$ даны. Однако, т.к. ***n*** не является простым числом, то решить такое сравнение можно только через систему сравнений по модулям множителей ***n***, но в нашем случае множители неизвестны. Поэтому попробуем расписать $\varphi(n)$ для упрощения решения.

$$\varphi(n) = (p-1)\cdot(q-1) = pq - p - q + 1 = n - p - q + 1$$
$$\varphi(n) - (p+q) + 1 \pmod{n}$$

- Обозначим $x = p + q$, тогда

$$3k1^2 \cdot \varphi^2(n) + 3k1 \cdot \varphi(n) + k1^3 \cdot \varphi_{enc} - 27d_{enc} + 1 \equiv 0 \pmod{n}$$

$$3k1^2 \cdot (1 - x)^2 + 3k1 \cdot (1 - x) + k1^3 \cdot \varphi_{enc} - 27d_{enc} + 1 \equiv 0 \pmod{n}$$

$$3k1^2 \cdot (1 - 2 \cdot x + x^2)^2 + 3k1 \cdot (1 - x) + k1^3 \cdot \varphi_{enc} - 27d_{enc} + 1 \equiv 0 \pmod{n}$$

$$3k1^2 \cdot x^2 + (-6k1^2 - 3k1)x + (3k1^2 + 3k1 + k1^3 \cdot \varphi_{enc} - 27d_{enc} + 1) \equiv 0 \pmod{n} (1)$$

- Далее зная что $x = p + q$, можем сделать вывод что x гораздо меньше n. Допустим p>q.

$$x^2 = (p + q)2 = p^2 + q^2 + 2 \cdot p \cdot q = p^2 + q^2 + 2n$$

- Зная что ***p***, ***q*** сгенерированы длиной 1024 бит, $p/q < 2$.
Можем записать такое неравенство:
$p \leq 2 \cdot q$, следовательно $p^2 \geq 4 \cdot q^2$. А также $q^2 \leq n$
$p^2 + q^2 + 2 \cdot n \leq 4 \cdot q^2 + q^2 + 2 \cdot n$
$x^2 \leq 5 \cdot q^2 + 2 \cdot n$
$x^2 \leq 5 \cdot n + 2 \cdot n$
$x^2 \leq 7 \cdot n$

- Используя коэффициент a из квадратного сравнения (1) распишем неравенство для x^2.
  $3k1^2 \cdot x^2 \leq 3 \cdot 2^2 \cdot x^2$, так как $k1 \leq 2$.
  $3k1^2 \cdot x^2 \leq 12 \cdot 7 \cdot n$
  $3k1^2 \cdot x^2 \leq 84 \cdot n$
  $3k1^2 \cdot x^2 + (-6k1^2 - 3k1)x + (3k1^2 + 3k1 + k1^3 \cdot \varphi_{enc} - 27d_{enc} + 1) < 84n$

- Мы знаем все числа из $\boldsymbol{b}$ и $\boldsymbol{c}$ коэффициентов неравенства выше, делаем вывод что $\boldsymbol{b}$ и $\boldsymbol{c}$ отрицательны, поэтому меняем знак на строгий.
  $3k1^2 \cdot x^2 + (-6k1^2 - 3k1)x + (3k1^2 + 3k1 + k1^3 \cdot \varphi_{enc} - 27d_{enc} + 1) = k2 \cdot n$, где $k2 \leq 84$, $k2 \in Z$.

- Таким образом мы можем перебрать все возможные $\boldsymbol{k}2$ чтобы решить уравнение, корнем которого будет $\boldsymbol{x}$, $x = p + q$.
- Для решения таска нам нужно знать $\varphi(n)$, оно находится с помощью выражения ниже, но мы также можем подсчитать p и q чтобы проверить наше решение $p \cdot q = n$. $\varphi(n) = n - p - q + 1$, домножим на p и перенесем всё в одну сторону.

$$pn - p^2 - pq + p - p \cdot \varphi(n) = 0$$
$$p^2 - pn - p + p \cdot \varphi(n) + n = 0$$
$$p^2 - p \cdot (n - \varphi(n) + 1) + n = 0$$

Решение задачи d-phi-enc

Теоретические рассчёты

- Решением данного квадратного уравнения будут p и q.
  Далее мы можем проверить выражение $p \cdot q = n$ и переходить к финальному шагу.
  Вычисляем $d = e^{-1} \pmod{\varphi(n)}$
  Вычисляем флаг: $flag = enc\_flag^d \pmod{n}$

- С помощью функции `long_to_bytes` вычисляем текстовое значение флага, которое равно:
  `b"HackTM{Have you warmed up? If not, I suggest you consider the case where e=65537, although I don't know if it's solvable. Why did I say that? Because I have to make this flag much longer to avoid solving it just by calculating the cubic root of enc_flag.}"`

Решение задачи d-phi-enc

Код для решения данной задачи

```python
from Crypto.Util.number import long_to_bytes

n = 2447638356...
enc_d = 2385197103...
enc_phi = 3988439673...
enc_flag = 2403368891
e = 3

for k1 in range(1, 3):
    a = 3*(k1^2)
    b = -(6*(k1^2)+3*k1)
    c = 3*(k1^2) + 3*k1 + (k1^3)*enc_phi - 27*int(enc_d) + 1

    #f = a*(x^2) + b*x + c
    det = b^2 - 4*a*c
    for k2 in range(85):
        c -= n
        det = b^2 - 4*a*c
        if(is_square(det)):break
    if(is_square(det)):break
```

Решение задачи d-phi-enc

Код для решения данной задачи

```
qrs_det = sqrt(b^2 - 4*a*c)
print("qrs_det=", qrs_det)

cand_x = (-1*b + qrs_det)/(2*a) #x=p+q
phi = n - cand_x + 1
print("phi=", phi)

p = ((n + 1 - phi) + sqrt(((n + 1 - phi) ^ 2) - 4 * n))/ 2
q = n // p
print("p: ", type(p), p)
print("q: ", type(q), q)
assert(p*q == n)

d = pow(e, -1, phi)
print("flag=", (pow(enc_flag, d, n)))
```

- Асимметричные криптографические системы основаны на так называемых односторонних функциях с секретом. Например, операция возведения числа в степень по модулю:

$$
\begin{aligned}
c &\equiv f(m) \equiv m^e \pmod{n} \\
m &\equiv f^{-1}(c) \equiv c^d \pmod{n} \\
d &\equiv e^{-1} \pmod{\varphi(n)}
\end{aligned}
$$

- Асимметричные криптографические системы основаны на так называемых односторонних функциях с секретом. Например, операция возведения числа в степень по модулю:

$$
\begin{aligned}
c &\equiv f(m) \equiv m^e \pmod{n} \\
m &\equiv f^{-1}(c) \equiv c^d \pmod{n} \\
d &\equiv e^{-1} \pmod{\varphi(n)}
\end{aligned}
$$

- ***c*** получено возведением в степень по модулю числа m. Назовём это действие шифрованием
  m - открытый текст, а ***c*** - шифртекст
  Пара чисел (***e***, ***n***) - открытый ключ
  d - закрытый ключ

- Пусть $\boldsymbol{n} = p \cdot q$, где где p и q – некоторые разные простые числа. Для такого n функция Эйлера имеет вид:

$$\varphi(n) = (p-1) \cdot (q-1)$$

Концепция RSA

Математическая модель

- Пусть $\boldsymbol{n} = p \cdot q$, где где p и q – некоторые разные простые числа. Для такого n функция Эйлера имеет вид:

$$\varphi(n) = (p-1)\cdot(q-1)$$

- Выберем такое число $\boldsymbol{e}$:

$$e \in [3, \varphi(n)-1]$$
$$\text{НОД}(e, \varphi(n)-1) = 1$$

Вычислим $\boldsymbol{d}$:

$$d \equiv e^{-1} \pmod{\varphi(n)}$$

- Возьмём в качестве сообщения число $m \in [1, n-1]$ Чтобы зашифровать его, необходимо возвести его в степень $\boldsymbol{e}$ по модулю $\boldsymbol{n}$

$$c \equiv m^e \pmod{n}$$

Отметим также, что $\boldsymbol{c} \in [1, n-1]$, как и $\boldsymbol{m}$. Расшифруем шифртекст, возведя его в степень закрытого ключа $\boldsymbol{d}$:

$$m` \equiv c^d \pmod{n}$$

Используемая литература

- Математика криптографии и теория шифрования. Лекция 13: Квадратичное сравнение. https://intuit.ru/studies/courses/552/408/lecture/9370
- RSA: от простых чисел до электронной подписи https://habr.com/ru/post/534014/
- Криптоанализ RSA. Сонг Ян, 2011 год. ISBN 978-5-93972-873-7.
- The CrypTool Book: Learning and Experiencing Cryptography with CrypTool and SageMath. Prof. Bernhard Esslinger and the Development Team of the Open-Source Software CrypTool. Edition 12 (2018). https://www.cryptool.org