

p4-team / ctf

Watch78

★ Star297

🍴 Fork62

<> Code

🔔 Issues 0

🔗 Pull requests 0

📁 Projects 0

🔍 Insights ▾


Branch: master ▾

ctf / 2016-11-05-hack-the-vote / hands_crypto_300 /



Create new file

Find file

History

 Pharisaeus added hands

Latest commit 27be70c on Nov 9 2016

..		
 README.md	added hands	7 months ago
 intercepted.txt	added hands	7 months ago

 README.md

Baby hands (crypto 300)

###ENG PL

We get some [data](#) with many triplets denoted as (d,c,n) . Our first idea was to check if maybe a pair of moduli share the same prime, but we found nothing. It was a bit confusing because there are not that many attacks which require many payloads.

And we were right - this one didn't. It needed only one payload, and each one gave the flag.

If we look at those payloads we can see that d is very very large, which actually suggest that the corresponding exponent e might be rather small. There is an efficient attack against large public exponent - Wiener attack, and it is exactly what we were supposed to do.

So we simply run:

```
# sage

n = 16237546855625534284018438001775230704957595514381112465166817954699914445541563226586260251438640941225877264379
e = 64193765095472280945778947695026260940793161700792092928929371930940586875921621250436677664062645637750266086941
c = 16136858024599713762543824813909888838980135983879214009979408405282927938342232267012266278670485820167254123223

c_fracs = continued_fraction(e / n).convergents()
test_message = 42
test_message_encrypted = pow(test_message, e, n)
for i in xrange(len(c_fracs)):
    if pow(test_message_encrypted, c_fracs[i].denom(), n) == test_message:
        d = c_fracs[i].denom()
        break
flag = pow(c, d, n)
print(flag)
```

And got `flag{G3t_1t?_1t_h4s_4_sm411_d}`

###PL version

Dostajemy [dane](#) z wieloma trójkami oznaczonymi jako (d,c,n) . Pierwszy pomysł to sprawdzenie czy nie ma pary modułów dzielących ten sam czynnik pierwszy, ale nic z tego. Było to trochę mylące, bo niewiele jest ataków które wymagają wielu wiadomości i kluczy.

I mieliśmy rację - ten atak wcale ich nie wymagał. Wymagał tylko jednej trójki i każda trójka dawała flagę.

Jeśli popatrzymy dokładnie na dane zauważymy że d jest bardzo bardzo duże, co sugeruje, że odpowiadający mu wykładnik e może być dość mały. Istnieje efektywny atak dla dużego wykładnika publicznego - atak Wienera i było to dokładnie to czego użyliśmy:

Uruchomiliśmy:

```
# sage

n = 16237546855625534284018438001775230704957595514381112465166817954699914445541563226586260251438640941225877264379
e = 64193765095472280945778947695026260940793161700792092928929371930940586875921621250436677664062645637750266086941
c = 16136858024599713762543824813909888838980135983879214009979408405282927938342232267012266278670485820167254123223

c_frac = continued_fraction(e / n).convergents()
test_message = 42
test_message_encrypted = pow(test_message, e, n)
for i in xrange(len(c_frac)):
    if pow(test_message_encrypted, c_frac[i].denom(), n) == test_message:
        d = c_frac[i].denom()
        break
flag = pow(c, d, n)
print(flag)
```

I dostaliśmy flag{G3t_1t?_1t_h4s_4_sm411_d}

