Features Business Explore Pricing

This repository Search

Sign in or Sign up

p4-team / ctf

Watch 74 Star 260 Fork 58

Code Issues 0 Pull requests 0 Projects 0 Pulse Graphs

Branch: master ctf / 2016-11-17-qiwi-2016 / dif-hel2 /

Create new file Find file History

Pharisaeus qiwi ctf writeups Latest commit 0cd3da1 on 12 Dec 2016

..

README.md qiwi ctf writeups 3 months ago

README.md

Diffie-Hellman 2 (crypto 300)

###ENG PL

In the task we get parameters for Diffie-Hellman key exchange. However, we get only the public secrets from the participants, like if we were sniffing them, and we want to get the shared secret from this.

We get:

```
p = 67039039649712985497870124991029230637396829102961966888617807218608820150367734884009371490834517138450159290932
ga = 84249833334845749358334422146936345855116076320439289003448782028 8 # g^a
gb = 89202980794122492566142873090593446023921664 # g^b
g = 9444732965739290427392
```

From this we want to calculate `g^ab mod p`, and therefore we need to know either `a` or `b`. Normally this would not be possible, but the numbers we have are rather small, so we can effecively compute a discrete logarithm and recover `a` from `g^a` or `b` from `g^b`:

```
def discrete_log(g, ga):
    for i in range(2, 1000):
        if g ** i == ga:
            return i


def main():
    p = 67039039649712985497870124991029230637396829102961966888617807218608820150367734884009371490834517138450159290932
    ga = 84249833334845749358334422146936345855116076320439289003448782028 8
    gb = 89202980794122492566142873090593446023921664
    g = 9444732965739290427392
    a = discrete_log(g, ga)
    b = discrete_log(g, gb)
    print(a)
    print(b)
    print(pow(gb, a, p))
    print(str(pow(gb, a, p))[:20])


main()
```

Which tells us that `a = 3`, `b = 2` and the flag is `70980344169492860405`

###PL version

W zadaniu dostajemy parametry dla wymiany klucza protokołem Diffiego Hellmana. Niemniej dostajemy jedynie publiczne sekrety uczestników, tak jakbyśmy ich sniffowali, a chcemy z tego uzyskać wspólny sekret.

Dostajemy:

```
p = 67039039649712985497870124991029230637396829102961966888617807218608820150367734884009371490834517138450159290932
ga = 842498333348457493583344221469363458551160763204392890034487820288 # g^a
gb = 89202980794122492566142873090593446023921664 # g^b
g = 9444732965739290427392
```

I z tego chcemy policzyć `g^ab mod p`, co oznacza że musimy poznać `a` lub `b`. Normalnie to nie byłoby możliwe, ale liczby na których operujemy są względnie małe więc możemy efektywnie policzyć logarytm dyskretny i odzyskać `a` z `g^a` lub `b` z `g^b`:

```
def discrete_log(g, ga):
    for i in range(2, 1000):
        if g ** i == ga:
            return i


def main():
    p = 67039039649712985497870124991029230637396829102961966888617807218608820150367734884009371490834517138450159295
    ga = 842498333348457493583344221469363458551160763204392890034487820288
    gb = 89202980794122492566142873090593446023921664
    g = 9444732965739290427392
    a = discrete_log(g, ga)
    b = discrete_log(g, gb)
    print(a)
    print(b)
    print(pow(gb, a, p))
    print(str(pow(gb, a, p))[:20])


main()
```

Co mówi nam ze `a = 3`, `b = 2` a flaga to `70980344169492860405`

© 2017 GitHub, Inc. Terms Privacy Security Status Help

Contact GitHub API Training Shop Blog About