

p4-team / ctf

Watch78

Star296

Fork62

<> Code

Issues 0

Pull requests 0

Projects 0

Pulse

Graphs

Branch: master

ctf / 2017-02-04-bitsctf / enigma /



Create new file

Find file

History

 Pharisaeus added enigma writeup

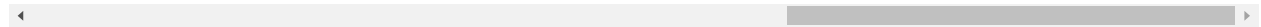
Latest commit 7a01f93 on Feb 5

..		
 README.md	added enigma writeup	3 months ago
 encrypted.tar.xz	added enigma writeup	3 months ago

 README.md

## Enigma (crypto)

ack. Sadly all we have got are the encrypted intercepted messages. Your task is to break the Enigma and get the flag.



###ENG PL

In the task we get a set of [ciphertexts](#) to work with. Initially we thought this is another one of repeating-key-xor and we were using our semi-interactive breaker for it, but it seemed to not work at all - we could not find any words. Then we decided to look at the data we got, and we saw for example:

```
Dtorouenc&Vguugaoct+Mihpio&dcuenksr|r&dco&06&Atgb&Hitbch&shb&73&Atgb&Qcurch(&Hcnkch&Uoc&cu&ui`itr
```

```
60<56*&Bgu&Qcrrct&our&ncsrc&mjgt(&Tcach&gk&Gdchb
```

What sticks of instantly is how many `&` are there. It can't be a coincidence so we figured that those have to be spaces and therefore the xor key has to be 1 or 2 characters at most. We checked and it turned out that it was a single `\6`.

We run:

```
import codecs
from crypto_commons.generic import chunk_with_remainder, xor_string

def main():
    cts = []
    for i in range(1, 7):
        with codecs.open("encrypted/" + str(i) + ".e", "r") as input_file:
            data = input_file.read()
            cts.append(data)
    xored = [xor_string(chr(ord('&') ^ ord(' ')) * len(data), d) for d in cts]
    print(xored)

main()
```

And we get `BITCTF{Focke-Wulf Fw 200}` in one of the messages.

###PL version

W zadaniu dostajemy zestaw [szyfrogramów](#). Początkowo myśleliśmy, że to kolejna wersja łamania powtarzającego się klucza xor i chcieliśmy użyć naszego semi-interaktywnego łamacza, ale nic ciekawego z tego nie wychodziło - nie mogliśmy znaleźć żadnych sensownych słów. Postanowiliśmy więc popatrzeć na dane które mamy w plikach:

Dtorouenc&Vguugaoct+Mihpio&dcuenksr|r&dco&06&Atgb&Hitbch&shb&73&Atgb&Qcurch(&Hcnkch&Uoc&cu&ui`itr

60<56\*&Bgu&Qcrrct&our&ncsrc&mjgt(&Tcach&gk&Gdchb

Co rzuca się od razu w oczy to liczba znaków & . To nie może być przypadek więc założyliśmy, że to mogą być spacje a tym samym klucz xora może mieć co najwyżej 1 lub 2 znaki. Sprawdziliśmy i okazało się że kluczem był znak \6 .

Uruchamiamy:

```
import codecs
from crypto_commons.generic import chunk_with_remainder, xor_string

def main():
    cts = []
    for i in range(1, 7):
        with codecs.open("encrypted/" + str(i) + ".e", "r") as input_file:
            data = input_file.read()
            cts.append(data)
    xored = [xor_string(chr(ord('&') ^ ord(' ')) * len(data), d) for d in cts]
    print(xored)

main()
```

I dostajemy BITCTF{Focke-Wulf Fw 200} w jednej z wiadomości.

