CHALLENGE : 192 Curves

CATEGORY : Crypto

AUTHOR : 0xsakthi

LEVEL : HARD

let us connect through the server!!

```
sakthi@debian:~/hacking/tamilctf-online$ nc 3.97.113.25 8001
~~ Server is working ~~
-- Karma says you get what you give, but not so fast --
** But give us what you will get, before you get what you want **
#hi
```

we want a flag okay!

```
sakthi@debian:~/hacking/tamilctf-online$ nc 3.97.113.25 8001
~~ Server is working ~~
-- Karma says you get what you give, but not so fast --
** But give us what you will get, before you get what you want **
#hi
Before Encrypt Message: hi
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e5244d0ec690a77f328ae3f04eb6f5d65e63b1d96891afb02b7
#tamilctf
Before Encrypt Message: tamilctf
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52d96664df9a89c687d1935479109e63af545a2f5ff5471261
#how
Before Encrypt Message: how
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e521ac2fd6253b847a244ebefba1f01f12ea8e378f0c6c4912c
#soahdaoioa
Before Encrypt Message: soahdaoioa
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52063dbba472e60f60146fc6269ca0b113dde8b6d763390cf6
#8291201
Before Encrypt Message: 8291201
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52f11656afb8f56fd3396c3dec2286d208561665bb798bd94e
#
```

when I enter a text it will be returned an encrypted hash, Notice the hash, The first part of the hash is the same for every text!!

Challenge name directly refers to its Elliptic Curve Cryptography!

we know the first part is the same for every text, so its reusing ***r*** //nonce are reused (critical vulnerability in ECC)

```
*The same random number r is used multiple times, this can be used to obtain the private key :) (nonces are reused)*
```

lets assingn the cipher value of flag(plain-text) and submit it via netcat server it will throw the flag //mentioned in discord(decoded qoate of nc banner :))

let us assign the value of the flag!

```
sakthi@debian:~/TamilCTF/Crypto-Writeups$ nc 3.97.113.25 8001
~~ Server is working ~~
-- Karma says you get what you give, but not so fast --
** But give us what you will get, before you get what you want **
#Tamil
Before Encrypt Message: Tamil
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e529bf70d1756b1c946246ab2be9ad13a5a869f5022a0b04230
#CTF
Before Encrypt Message: CTF
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e523cd5fd82c12d544691f80dd98e3d5bd9f4893c12ac976bf7
#
```

Tamil

fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e529bf70d1756b1c946246ab2be9ad13a5a869f5022a0b04230

Ctf

fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e523cd5fd82c12d544691f80dd98e3d5bd9f4893c12ac976bf7

script to assign the value of the flag!

```python
#!/usr/bin/python3
from hashlib import sha1
from ecdsa.curves import NIST192p
from ecdsa.numbertheory import inverse_mod
from ecdsa import SigningKey
import hashlib

n = NIST192p.order
r1 = int('fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52',16)
s1 = int('9bf70d1756b1c946246ab2be9ad13a5a869f5022a0b04230',16)
m1 = b'Tamil'
z1 = int(hashlib.sha1(m1).hexdigest(),16)
#print(z1)
r2 = int('fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52',16)
s2 = int('3cd5fd82c12d544691f80dd98e3d5bd9f4893c12ac976bf7',16)
m2 = b'CTF'
z2 = int(hashlib.sha1(m2).hexdigest(),16)
#print(z2)
k = (((z1 - z2) %n) * inverse_mod(s1-s2,n)) % n
print('k values is:',k)
dA = ((((s1 * k) % n) - z1) * inverse_mod(r1, n)) %n
print('dA value is :',dA)
sk = SigningKey.from_secret_exponent(dA)
sig = sk.sign(b'flag', k=k)
print('Binary Value:',sig)
print('Assingned Msg:'+ sig.hex())
```

Detail About The Script

```
#Tamil
Before Encrypt Message: Tamil
After Encrypt Message:

fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52 === r1
9bf70d1756b1c946246ab2be9ad13a5a869f5022a0b04230 === s1(msg)

#CTF

Before Encrypt Message: CTF
After Encrypt Message:

fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52 === r2
3cd5fd82c12d544691f80dd98e3d5bd9f4893c12ac976bf7 === s2(msg)

r1==r2
```

```
n = NIST192p.order
r1 = int('fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52',16)
s1 = int('9bf70d1756b1c946246ab2be9ad13a5a869f5022a0b04230',16)
m1 = b'Tamil'
z1 = int(hashlib.sha1(m1).hexdigest(),16)
#print(z1)
r2 = int('fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e52',16)
s2 = int('3cd5fd82c12d544691f80dd98e3d5bd9f4893c12ac976bf7',16)
m2 = b'CTF'
z2 = int(hashlib.sha1(m2).hexdigest(),16)
```

n is a curve order, r1 and r2 are the same,s1 and s2 our msg(encrypted form),m1 and m2 is our message(plain-text)!
m1 and m2 are bytes we need to do calculation so I convert the bytes to sha1 stored in z1 and z2

```
k = (((z1 - z2) %n) * inverse_mod(s1-s2,n)) % n
print('k values is:',k)
```

> As the standard notes, it is not only required for k to be secret, but it is also crucial to select different k for different signatures, otherwise the equation in step 6 can be solved for d_A, the private key: given two signatures (r, s) and (r, s'), employing the same unknown k for different known messages m and m', an attacker can calculate z and z', and since $s - s' = k^{-1}(z - z')$ (all operations in this paragraph are done modulo n) the attacker can find $k = \frac{z - z'}{s - s'}$. Since

formula to recover 'K'

```
why I mentioned %n in every calculation?
```

because n is an order of curve, we are doing the calculations in the **'curve'** ...

```
dA = ((((s1 * k) % n) - z1) * inverse_mod(r1, n)) %n
print('dA value is :',dA)
```

$$d_A = \frac{sk - z}{r}.$$

formula to obtain the private key!
lets assign the value

```
sk = SigningKey.from_secret_exponent(dA)
sig = sk.sign(b'flag', k=k)
```

Result:

```
sakthi@debian:~/TamilCTF/Crypto-Writeups$ python3 flag.py
k values is: 2021
dA value is : 49
Binary Value: b'\xfc0\xdd?\xfa-\xdc\xbf%5\xd4\x99\x84\xa5Y2\x8f\xb3\xb06\xb8\x84\x9eR9\xcb\xba\xe1\xe2:\x02\x98\'\x7f\x88\x97\xb4"\xe7\xc1\x02El\xcf\\v\xd3\xbd'
Assingned Msg:fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e5239cbbae1e23a0298277f8897b422e7c102456ccf5c76d3bd
sakthi@debian:~/TamilCTF/Crypto-Writeups$
```


```
sakthi@debian:~/TamilCTF/Crypto-Writeups$ python3 flag.py
k values is: 2021
dA value is : 49
Binary Value: b'\xfc0\xdd?\xfa-\xdc\xbf%5\xd4\x99\x84\xa5Y2\x8f\xb3\xb06\xb8\x84\x9eR9\xcb\xba\xe1\xe2:\x02\x98\
xb4"\xe7\xc1\x02El\xcf\\v\xd3\xbd'
Assingned Msg:fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e5239cbbae1e23a0298277f8897b422e7c102456ccf5c76d3bd
sakthi@debian:~/TamilCTF/Crypto-Writeups$
```


value of flag in dA=49 k=2021

fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e5239cbbae1e23a0298277f8897b422e7c102456ccf5c76d3bd

let's submit the value in netcat server!

```
-- Karma says you get what you give, but not so fast --
** But give us what you will get, before you get what you want **
```

```
sakthi@debian:~/TamilCTF/Crypto-Writeups$ nc 3.97.113.25 8001
~~ Server is working ~~
-- Karma says you get what you give, but not so fast --
** But give us what you will get, before you get what you want **
#Tamil
Before Encrypt Message: Tamil
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e529bf70d1756b1c946246ab2be9ad13a5a869f5022a0b04230
#CTF
Before Encrypt Message: CTF
After Encrypt Message: fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e523cd5fd82c12d544691f80dd98e3d5bd9f4893c12ac976bf7
#fc30dd3ffa2ddcbf2535d49984a559328fb3b036b8849e5239cbbae1e23a0298277f8897b422e7c102456ccf5c76d3bd
TamilCTF{Reu$iNg_is_D4Ng3r$_in_3Rypt0}
```

Flag

```
TamilCTF{Reu$iNg_is_D4Ng3r$_in_3Rypt0}
```

First Blood : DarkArmy