Round Rabins (70 p)

Breaking Rabin cryptosystem is hard if the primes were chosen properly. This is probably the flaw here, or the challenge would be computationally hard. Lets try factordb.com. It reports that `N` is square. OK, great.

```
import libnum

N = 0x6b612825bd7972986b4c0ccb8ccb2fbcd25fffbadd57350d713f73b1e51ba9fc4a6ae862475efa3c9fe7dfb4c89b4f92e925ce8e8eb8af1
c = 0xd9d6345f4f961790abb7830d367bede431f91112d11aabe1ed311c7710f43b9b0d5331f71a1fccbfca71f739ee5be42c16c6b4de2a9cbee

x = libnum.common.nroot(N, 2)
assert(N == x ** 2)
```

The code passes, so we are fine. Now, how do we solve a modular square root in squared prime modulus x^2? First of all, we can solve the simpler problem in the smaller field Z_x. We can use for instance PARI/GP `factor(x^2 - Mod(c%p,p))` . We now have the square roots

```
m1 = 11979941539608683221717291954593074711590148397596506725379995777962253281877636373276686297362111446138893316731
m2 = p - m1
```

We now need to lift it to square modulus, i.e., m_1 mod x^2. We achieve this as follows

```
q = (c - m1 ** 2) / p
l = q * libnum.modular.invmod(2 * m1, p)
m = m1 + l * p

print libnum.n2s(m % N)
```

Running this, we get the flag

```
IceCTF{john_needs_to_get_his_stuff_together_and_do_things_correctly}
```

Contract (130 p)

```
Our contractors stole the flag! They put it on their file server and challenged us to get it back. Can you do it for
```

This is clearly a nonce reuse, which leads to a standard attack. First, we compute the secret value $k = (z_1 - z2) \times (s_1 - s2)^{-1}$ using a signature pair. Then, using a single signature in conjunction with k, we may find $d = (s_1 \times k - z_1) \times (r_1)^{-1}$. All modular operations are performed mod n. Embodied in Python, the attack is performed as follows.

```
import hashlib, libnum, binascii, socket
from ecdsa import VerifyingKey, SigningKey

def send(message):
    s = socket.create_connection(('contract.vuln.icec.tf', 6002))
    s.send(message + '\n')
    print s.recv(1024)
    print s.recv(1024)
    return

PUBLIC_KEY = '''
-----BEGIN PUBLIC KEY-----
MHYwEAYHKoZIzj0CAQYFK4EEACIDYgAEgTxPtDMGS8oOT3h6fLvYyUGq/BWeKiCB
sQPyD0+2vybIT/Xdl6hOqQd74zr4U2dkj+2q6+vwQ4DCB1X7HsFZ5JczfkO7HCdY
I7sGDvd9eUias/xPdSIL3gMbs26b0Ww0
-----END PUBLIC KEY-----
'''

vk = VerifyingKey.from_pem(PUBLIC_KEY.strip())
n = vk.pubkey.order

help_cmd = 'help:c0e1fc4e3858ac6334cc8798fdec40790d7ad361ffc691c26f2902c41f2b7c2fd1ca916de687858953a6405423fe156cfd72
time_cmd = 'time:c0e1fc4e3858ac6334cc8798fdec40790d7ad361ffc691c26f2902c41f2b7c2fd1ca916de687858953a6405423fe156c0cbe
read_flag_cmd = 'read flag.txt'
```