# Author: Trevor Perrin

# See the LICENSE file for legal information regarding use of this file.

"""PyCrypto RSA implementation."""

from .cryptomath import \*

from .rsakey import \*

from .python\_rsakey import Python\_RSAKey

if pycryptoLoaded:

from Crypto.PublicKey import RSA

class PyCrypto\_RSAKey(RSAKey):

def \_\_init\_\_(self, n=0, e=0, d=0, p=0, q=0, dP=0, dQ=0, qInv=0):

if not d:

self.rsa = RSA.construct( (long(n), long(e)) )

else:

self.rsa = RSA.construct( (long(n), long(e), long(d), long(p), long(q)) )

def \_\_getattr\_\_(self, name):

return getattr(self.rsa, name)

def hasPrivateKey(self):

return self.rsa.has\_private()

def \_rawPrivateKeyOp(self, m):

c = self.rsa.decrypt((m,))

return c

def \_rawPublicKeyOp(self, c):

m = self.rsa.encrypt(c, None)[0]

return m

def generate(bits):

key = PyCrypto\_RSAKey()

def f(numBytes):

return bytes(getRandomBytes(numBytes))

key.rsa = RSA.generate(bits, f)

return key

generate = staticmethod(generate)