hashlib

--hash = hashlib.md5()/sha1()/sha256()/sha384()/sha512()

采用相应的算法进行加密

--hash.update(s.encode('utf8'))

如果同一个hash对象重复调用该方法，则m.update(a); m.update(b) is equivalent to m.update(a+b).

--hash.hexdigest()

返回摘要，作为十六进制数据字符串值

--hash.digest()

返回摘要，作为二进制数据字符串值

FE：

```
>>> s = hashlib.md5('efeihfiuewhfuihiwu'.encode('utf8'))
>>> s
<md5 HASH object @ 0x000000F5E33EEEE0>
>>> s.digest()
b'X\xac\xbb\x93,}\x9f\x83\x16\xdb\x05\x012\r\x86r'
>>> s.hexdigest()
'58acbb932c7d9f8316db0501320d8672'
```

--hash.digest_size

产生的散列的字节大小

--hash.block_size

The internal block size of the hash algorithm in bytes.

--n = hashlib.new(name,string.encode('utf8'))

创建new对象，name是算法名字

FE：

```
h2 = hashlib.new('ripemd160','what'.encode('utf8'))
```