**实验原理：**

Sm3算法首先要将长为L的消息m进行填充，首位用1 填充，其余位用K个0填充，填充后的消息长度L+1+K=448（mod512）,最后用64比特表示L。然后将消息m按512比特分组B[0],B[1],B[2]…然后要对消息进行压缩处理，V[i+1]=CF(V[i],B[i])，初始V[0]为常量IV。最后得到的V[i+1]即为结果。

假设有未知的secret，长度为3，那么将要对其补充strb=x80+x00\*59+x18，补充后的消息为strb1=secret+strb。其生成的sm3摘要为Digest0。

构造str1=strb+strc。那么算法将会将str1进行分组，第一组为strb,然后用第一组的结果对strc进行计算，最终得到有Digest1

那么在不知道secret的情况下，设有长度为3的消息strx,同样对其补充strb=x80+x00\*59+x18,补充后的消息为strb2=strx+strb。构造str2=strb2+strc，算法会将消息分为两组，第一组为strb2,第二组为strc,然后用第一组的结果对第二组进行计算，此时修改初始V[0],将初始V[0]设为Digest1,用V[0]直接对strc进行计算，可以得到Digest2,Digest2=Digest1。即在不知道secret的情况下也可以通过长度扩展攻击得到同样的sm3消息摘要。

**实验过程：**

在本实验中设立secret为‘\x61\x62\x63’,生成

1.Digest0=66c7f0f462eeedd9d1f2d46bdc10e4e24167c4875cf2f7a2297da02b8f4ba8e0

对应的V[1]= [1724379380, 1659825625, 3522352235, 3692094690, 1097319559, 1559426978, 696098859, 2404100320]

2.设strc=‘\x61\x62\x63’,则str1=’\x61\x62\x63’+ ‘\x80+\x00\*59+\x18’+’\x61\x62\x63’,生成的消息摘要为accf73ffc14f32535d8814377cf8cad2238c905fbdb50528036f5bef72052639

。

3.任意输入三位长度的strx,假设strx=’fsw’,则str2=’fsw’+ ‘\x80+\x00\*59+\x18’+’\x61\x62\x63’，修改sm3的算法代码，使初始IV=V[1](V[1]为第一步中的V[1]),并在进行压缩函数的循环函数时修改使初始从B[1]计算，即直接计算V[2]=CF(V[1],B[1]),生成的消息摘要为accf73ffc14f32535d8814377cf8cad2238c905fbdb50528036f5bef72052639

**实验中遇到的问题：**

1 要将消息转换为Bytes，但在使用encode函数进行转换时，对’\x80‘使用encode函数得到b'\xc2\x80'，莫名其妙出现了\xc2，造成结果有问题。解决方法：直接使用b’\x80’。

2 将消息摘要Digest0转换成V[1]时，直接将整体进行了进制转换，导致结果错误。解决方法：将Digest0分成8组，分别对其进行进制转换。

3 直接修改了初始IV，没有修改压缩函数中的循环代码，导致结果出错。解决方法：修改压缩函数代码中的循环部分。