Project3: implement length extension attack for SM3, SHA256, etc.

实现SM3, SHA256等的长度扩展攻击

**SM3:**

长度扩展攻击是一种针对哈希算法的攻击方法，利用哈希函数的特性，在已知哈希值的情况下，构造出具有相同哈希值但内容不同的消息。下面是对上述代码中长度扩展攻击方法及原理的详细讲述：

1. 首先，我们需要了解SM3哈希算法的一些特性。SM3是一种密码杂凑算法，它将输入消息分成512位的块，并对每个块进行一系列的操作，最终生成256位的哈希值。SM3算法使用了一系列的位运算和非线性函数来处理消息块。

2. 在长度扩展攻击中，我们假设攻击者已经获取到了原始消息的哈希值，并希望构造出一个具有相同哈希值但内容不同的消息。

3. 首先，我们需要构造一个填充函数，该函数将根据原始消息的长度生成正确的填充，以便使填充后的消息长度满足SM3算法的要求。填充函数`sm3\_padding`接受原始消息作为输入，并生成填充后的消息。

4. 接下来，我们使用原始消息调用`sm3\_hash`函数来计算原始消息的哈希值。这里使用了Python的`hashlib`库来计算SM3哈希值。

5. 在长度扩展攻击中，我们还需要提供扩展数据，即我们想要附加到原始消息后面的内容。

6. 然后，我们使用填充函数生成填充后的原始消息，并获取原始消息的长度。

7. 接下来，我们将原始哈希值解析为8个32位整数的列表，表示SM3算法的内部状态。

8. 我们使用填充后的原始消息更新内部状态。首先，我们使用填充后的原始消息更新内部状态的前4个整数，通过异或操作与固定的常量进行运算。然后，我们使用填充后的原始消息更新内部状态的后4个整数，通过异或操作与扩展数据的哈希值的对应整数进行运算。

9. 最后，我们将更新后的内部状态打包为字节串，并得到新的哈希值。

10. 最后，我们打印原始消息的哈希值和构造出的新消息的哈希值，以验证长度扩展攻击的成功。

总结来说，长度扩展攻击利用了哈希函数的特性，通过构造填充后的原始消息和扩展数据的哈希值，以及对内部状态的适当修改，成功生成了具有相同哈希值但内容不同的消息。这种攻击方法在实际应用中是非法的，并且SM3算法目前没有已知的实际攻击。在使用哈希算法时，请遵循安全实践，并使用经过广泛接受的标准算法。

**SHA256:**

长度扩展攻击是一种针对哈希函数的攻击方法，利用了哈希函数的特性来伪造具有相同哈希值的消息。在这种攻击中，攻击者能够根据已知的哈希值和消息的部分内容，计算出附加内容的哈希值，而无需知道附加内容的实际值。

下面是对长度扩展攻击方法及原理的详细解释：

1. 哈希函数选择：首先，攻击者需要了解目标系统使用的哈希函数。在这个例子中，我们选择了SHA-256作为目标哈希函数。

2. 原始消息和哈希值：攻击者需要获取原始消息和其对应的哈希值，这些信息可以通过不同的方式获取，例如通过网络传输、数据库泄露等手段。

3. 哈希函数工作原理：理解哈希函数的工作原理对于实施长度扩展攻击非常重要。SHA-256使用Merkle-Damgard结构，将消息分成固定大小的块，并对每个块进行处理。在处理过程中，哈希函数维护一个内部状态，称为哈希值。攻击者需要了解哈希函数的内部状态和填充规则。

4. 填充和长度：在长度扩展攻击中，攻击者需要了解原始消息的长度以及SHA-256的填充规则。SHA-256将消息填充为满足长度要求的倍数，并附加一个表示原始消息长度的位数。

5. 伪造消息构造：攻击者根据原始消息、填充规则和附加数据构造伪造消息。伪造消息由原始消息、填充和附加数据组成。

6. 伪造哈希值计算：攻击者使用SHA-256算法对附加数据进行哈希计算，得到附加数据的哈希值。

7. 伪造消息和哈希值：最后，攻击者将伪造消息和伪造哈希值作为攻击结果返回。

在上述代码中，`sha256\_length\_extension\_attack`函数实现了长度扩展攻击。它通过获取原始哈希值的状态、计算填充位数、构造伪造消息和计算伪造消息的哈希值来实现攻击。最终，函数返回伪造的消息和哈希值。

需要注意的是，长度扩展攻击是一种危险的攻击技术，仅用于教育和研究目的。在实际应用中，哈希函数应该与消息认证码（MAC）或其他安全机制结合使用，以防止此类攻击。