联邦学习通过加密机制下的参数交换方式保护用户数据隐私，其中一个重要手段便是同态加密。在前文中，我们已经提到了RSA算法与Paillier 算法，它们对同态加密的发展都起到了至关重要的作用。但RSA算法仅对乘法同态，Paillier算法仅对加法同态，它们都仅是部分同态。

Boneh-Goh-Nissim方案确实既满足加法同态又满足乘法同态，但随着计算的进行，噪音会不断增长，尤其是在进行乘法时急剧增长。待噪音增长到一定程度，便会超过计算机限制、出现奇偶翻转。这也就意味着，算得的密文无法被解密，也就无法保证计算结果的正确性了。因此它也称为有限次的全同态。

既然噪音会不断增长，那就算几轮就更新一次密文，使得密文中的噪音降为0，然后继续用来计算，如此循环往复0。但我们当然不可能把数据解密后重新加密。2009年，Craig Gentry提出了一个办法，叫做Bootstrapping，很好地解决了这一问题。下面，我们将阐述Bootstrapping办法。

Gentry的创新之处在于让解密过程本身也在密文情况下进行，也就是说，把解密过程当成是一种密文状态下的用来清除噪音的计算，在这一过程中其他人也无法得知明文内容。我们假设要对明文a进行加密，共需要2个秘钥对。首先用秘钥对[1]将a加密成b1，此时 刚刚被转化过来的b1噪音含量很低，可以进行数次计算。在多次计算过后得到b2，如果此时b2的噪音含量已经达到一定程度，就要启动Bootstrapping过程。先用密钥对[2]将b2加密成为c0、将密钥对[1]加密成密钥对[1’]，再用密钥对[1’]将c0解密成c1，由于此时密钥对[1’]在密钥对[2]的加密中，密文c0在密钥对[1]与密钥对[2]的双重加密中，所以解得的c1仍然是密文，且消除了密钥对[1]的加密带来的噪音，可以继续计算。在噪音达到一定程度后，可以用密钥对[1]对密钥对[2]做相同操作，消除噪音的同时将密文转化为由密钥对[1]加密。周而复始，第一次实现了真正的全同态加密。

Bootstrapping方法的缺陷同样不可忽视。由于它是对Boneh-Goh-Nissim方案的改进，仍然受到计算次数的限制。同时，由于解加密需要占用不小的空间，所以每轮计算的最后需要保留解密的空间，导致计算次数减少，解密次数增多。而解密的步骤往往也较为复杂，这就导致此方法计算量很大，效率低下。

尽管Gentry的办法不是尽善尽美，但它向世人展示了全同态加密的可行性，具有重要意义。