GCM Cipher Suites with Extended Packet Numbering

Mick Seaman[1]

This note is a first step towards a proposal for a GCM-based 802.1AE (MACsec) Cipher Suite that allow more than 2^{32} packets to be sent with a single Secure Association Key (SAK). GCM–AES–128 (the Default Cipher Suite) and GCM–AES–256 (its 256-bit keyed equivalent) use a 32-bit packet number (PN) as part of the IV (Initial Value) that has to be unique for all packets sent with a given SAK. All the participants in a given secure Connectivity Association can be using the same SAK and the remainder of the 96-bit IV is a 64-bit SCI that is guaranteed to differ for each participant.

The existing MACsec Cipher Suites and the SecTAG added to each MACsec protected packet were carefully chosen to have broad applicability, particularly across the wide speed range of current and future transmission speeds. However some consider the 2^{32} single SAK packet limit a problem. At 400 Gb/s, with minimum (Ethernet) packet sizes and full line utilization, an SAK can be exhausted in a little over 2 seconds. While the MACsec Key Agreement (MKA) protocol (specified in 802.1X-2010) can refresh SAKs at that rate, the timing is definitely tight. Data plane forwarding may not be maintained while MKA operation is suspended during in-service software upgrades (~60 seconds, risking PN exhaustion with more realistic packet sizes and line utilization).

This note uses GCM–AES–XPN as a working name for an extended PN proposal, presuming that both GCM-AES-XPN-128 and GCM-AES-256 Cipher Suite variants may be desirable. Since a conformant GCM-AES-XPN system using would also have to implement the existing GCM-AES-128, it is clearly desirable to maximise commonality with the existing MACsec specification. This proposal retains the existing SecTAG and data packet format and the existing management/MIB, and borrows much of the existing Cipher Suite detail. Design choices were significantly informed by SP 800–38D's requirement[2] for a 96-bit IV when more than 2^{32} invocations are made with a single key. This rules out 'simply extending' the PN field in the SecTAG.

In brief GCM–AES–XPN uses an IV derived from a 32-bit 'Short SCI'[3] (SSCI) and a 64-bit 'eXtended Packet Number' (XPN). The SSCI is assigned by the Key Server in such a way that no two participants can use the same SAK with the same SSCI. Each data packet carries the existing SCI (explicitly or implicitly, as per the current specification) and is used to find (index) the SSCI as well as the existing operational and management variables. The XPN comprises the 32-bit PN field carried in the SecTAG (as at present) plus 32 more-significant (upper) bits. Whenever a fresh SAK is distributed each participant sets the upper bits for that key, for itself and for all other participants, to zero. Increments to the upper bits are driven by the receipt of data (see below).

GCM–AES–XPN, as proposed, leans more heavily on MKA than would have been acceptable when 802.1AE was first standardized. MKA has to be extended, if only in minor ways. However some supporting changes are inevitable, if only to avoid declaring a system 'not live' when it suspends MKA operation for an in-service upgrade. This note describes the necessary amendments for both 802.1X and 802.1AE.

In addition to the changes described above GCM-AES-XPN incorporates GCM related improvements developed since the time of the original 802.1AE specification. These

[1]With thanks to Brian Weis for discussion of these issues, though all errors are my own. This version of this document is a minor revision of one privately circulated in July 2011, I have updated it for posting to the 802.1 server because it mentions a few things that may not be apparent from other documentation.

[2]SP 800–38D November 2007 Section 8.3 '***Constraints on the Number of Invocations***' "all implementations ... with IVs whose length is not 96 ... **total number of invocations of the authenticated encryption function shall not exceed 2^{32}** .."

[3]I am not the first person to propose this construct. In RFC 4106 (GCM ESP), the GCM IV (referred to as the nonce in that specification) comprises a 32-bit value (called the 'Salt' in that specification) concatenated with a 64-bit ESP,.

includes a per SAK 'Salt' proposed by David McGrew[1]. This 96-bit Salt is XOR'd with the concatenated SSCI and XPN to yield the IV, so the IV's for data packets transmitted by any given packets are no longer comprise a simple incrementing sequence.

An alternative to the current proposal would have been to borrow a number of bits, possibly all 16, from the notional port number field. I am not suggesting this, for a number of reasons, including the following. Effectively making the SCI a 48-bit MAC Address is reasonably likely to encourage the wasteful allocation of 'just in case' addresses, as the 'ports' at which MACsec can be used are not limited to the physical ports of a system. There are already cases in which MAC Addresses are wastefully allocated to support various forms of virtualization, and some of those doing so have been hostile to change. In such an environment a requirement for spare SCI's might simply deduct a further bit or two from the total number of addresses available. In the end this problem is only likely to be solved by a wholesale move (already advocated by the RAC) from 48-bit addresses to 64-bit EUIs, which would bring us back to the existing 64-bit SCI if 802.1AE is still to be relevant.

[1] draft-mcgrew-iv-gen-01.txt (reference updated February 2012).