11 Security-oriented mechanisms

11.7 Cryptographic method

11.7.4 Initialization Vector (IV) construction

11.7.4.1 Cipher clock

11.7.4.1.1 Cipher clock alignment in the upstream

11.7.4.1.2 Cipher clock alignment in the downstream

11.7.4.1.3 Initial cipher clock synchronization

While the OLT and ONU MPCP clocks are synchronized as part of the MPCP Discovery and Registration process (see IEEE Std 802.3, 144.3.1.1), the *RxCipherClock* and the *TxCipherClock* require an additional synchronization procedure to synchronize the 16 most-significant bits.

To initiate the cipher clock synchronization at the ONU, the OLT issues a *Set_Request* OAMPDU containing the *Sync Cipher Clock* TLV (see 14.6.5.2). The OLT shall not activate the initial encryption key for an ONU until it receives a positive acknowledgement from that ONU that the *Sync Cipher Clock* TLV was processed successfully.

The OLT shall form the *Sync Cipher Clock* TLV by setting its *RxCipherTimestamp* and *TxCipherTimestamp* fields as follows:

```
RxCipherTimestamp = CipherClock;
TxCipherTimestamp = RxCipherTimestamp + RTT[PLID];
```

The `RTT[PLID]` is the round-trip time value measured for the given ONU (PLID) at the time of its MPCP discovery. Note that adding `RTT[PLID]` may cause the value `TxCipherTimestamp` to wrap around.

It may not be possible to tightly control the transmission time of OAMPDUs, unlike that of MPCPDUs. It is acceptable for the OAMPDU containing the *Sync Cipher Clock* TLV to be transmitted after the time epoch corresponding to the captured value of the OLT's *CipherClock*, but the transmit time (referenced to the ESH of the envelope containing this OAMPDU) shall not lag behind the said time epoch by more than 1 second.

When the ONU receives the *Sync Cipher Clock* TLV, it increments both the `TxCipherTimestamp` and the `RxCipherTimestamp` until the bottom 32 bits of the `TxCipherTimestamp` match the current value of its local MPCP clock (i.e., `LocalTime` variable). The amount of such increment depends on how much the transmission time of *Sync Cipher Clock* TLV lagged behind the captured value of OLT's *CipherClock*. Note that incrementing the `TxCipherTimestamp` or the `RxCipherTimestamp` may cause the values to wrap around.

```
while( TxCipherTimestamp[31:0] != LocalTime)
{
   TxCipherTimestamp ++;
   RxCipherTimestamp ++;
}
```

Once the bottom 32 bits of `TxCipherTimestamp` match the `LocalTime`, the values of *TxCipherClock* and *RxCipherClock* are set by writing the corresponding adjusted timestamps into the *acSyncCipherClock* attribute (see 14.6.5.2):

```
acSyncCipherClock.sTxCipherClock = TxCipherTimestamp;
acSyncCipherClock.sRxCipherClock = RxCipherTimestamp;
```

From this moment on, the *TxCipherClock* and the *RxCipherClock* increment synchronously with the ONU's MPCP clock.

If the *TxCipherClock* and the *RxCipherClock* are synchronized properly, the following holds true:

— The bottom 32 bits of *TxCipherClock* match the local MPCP time at all times (`TxCipherClock[31:0] == LocalTime`).

— The bottom 6 bits of *RxCipherClock* match the value of the EPAM field in any received envelope header.

Implementations may choose to verify the above conditions in order to ensure proper *RxCipherClock* and *TxCipherClock* alignment.