End to End Data Protection with Encryption

David Kwak; dkwak@marvell.com
Ali Khwaja; akhwaja@marvell.com
Craig W. Carlson; cwcarlson@marvell.com

T11-2021-00021-v000
Goal

- Provide uninterrupted end-to-end data protection
  - Never allow a window where undetected corruption can take place
  - Focus of this presentation is the encryption/decryption phase for FC-SP-2 based encryption

- Encryption protection
  - Window of risk during encryption/decryption process
    - An error in the crypto engine can result in undetected corrupted data
    - This error could be the result of a memory error or other source
During Encryption process, FC Frame CRC recomputed on encrypted data
- This leaves a window of vulnerability during which errors could be introduced
- An error during the encryption process could introduce undetected data corruption
- This violates the premise of end-to-end data protection
- (NOTE: The same type of vulnerability also exists during decryption)

NOTE: ICV (Integrity Check Value) in the ESP_Trailer does not cover this
- Computed AFTER payload is encrypted
Why does T10 DIF or NVMe Protection information not cover this?

- T10 DIF and NVMe Protection Information are an FC-4 feature
  - Not under control of the Fibre Channel transport
    - May or may not be implemented or enabled by the devices

- The goal is to have **Transport** level end-to-end protection
FC ESP Double Encryption Process

- One solution
  - Encrypt the data twice and compare the results
  - If CRCe1 and CRCe2 are equal then data integrity has been maintained and encrypted frame can be transmitted
  - This would also be done on the receiver side as a double decryption

- Downside to this approach
  - Added complexity and possible extra latency to encryption/decryption process
  - To do efficiently, may require dual encryption/decryption engines

- Pro
  - This approach does not require any change to the Standard
Proposed Solution

- Propose adding an optional 16-bit CRC field to the ESP Padding
  - CRC would be computed before encryption takes place on the transmitter and checked after decryption on the receiver
  - Allows for overlapping data protection with FC Frame CRC (or other data checking as done by implementation)
  - Proposing basing CRC on T10 DIF CRC

- Pros
  - Does not require double encryption/decryption
    - Instead simple fast CRC calculation

- Cons
  - Requires Standard change
    - But, if we make it optional/negotiable allows for older implementations
Overlapping protection
- On Transmit
  - T10 CRC computed
  - Data encrypted
  - FC CRC recomputed as FC CRCe
- On Receive
  - FC CRCe checked
  - Data decrypted
  - Decrypted data checked against T10 CRC

Note: Implementations may not present full FC Frames from/to the encryption engine, but this mechanism works even if there is simply a check on the data going into/out of the encryption engine
- What matters is that T10 CRC checks that data was not corrupted during encryption/decryption
Frame Format Proposal

- Add 16 bit T10 DIF CRC to beginning of ESP padding
  - 16 bits is the minimum pad length

- T10 DIF CRC is calculated across
  - Unencrypted payload + Other optional headers
  - FC Frame header with D_ID, S_ID, and CS_CTL set to zero
    - This is the same as done for the Integrity Check Value in the ESP_Trailer
Other Details

- Other details to be determined
  - Negotiation on support of use of T10 DIF CRC in ESP_Trailer padding

- Potentially allow CRC negotiation to support newer 32 bit and 64 bit DIF defined in T10
Summary

- End-to-end data protection is important for enterprise customers

- Proposal is to allow for a standardized, optional, backward compatible data check field on the encrypted data
  - By adding CRC field to ESP padding

- Thoughts?
Questions from previous T11 meeting

- Is there a 16-byte (128-bit) TX alignment requirement?
  - No

- ESP Trailer Pad Length: Does the value include the CRC or not?
  - Yes, the CRC just replaces the first 2 bytes of the Pad

- ESP Trailer Pad Pattern/Sequence: following CRC – does it start at 0x1 or 0x3?
  - 0x3 – The CRC just replaces the first 2 bytes of the Pad

- Other Encryption Methods supported? Such as ENCR_NULL_AUTH_AES_GMAC? Not encrypted but with ESP_Header and ESP_Trailer.
  - Other methods could be supported, but they are not part of our current proposal - Should they be?
More questions for previous T11

- There are other alternatives without modifications to the standard – Implement a protected engine – a multitude of different options
  - Maybe, but that’s not our proposal since it doesn’t give you an interoperable end-to-end protection

- Why T10 CRC? Why not FC CRC?
  - We use the T10 CRC because it is 16 bits and that will fit in the minimum PAD length (thus, no changes to payload size)
  - Any other CRC algorithms (i.e., more BITS) would require more complicated changes

- Interesting: FC Frame header in CRC but ESP Header and Encryption IV is not.
  - The CRC calculation is done on the same fields as the ICV – This simplifies deployment

- If We’re Changing The Standard… – Option to expose additional parts of the original frame in clear text – Example: Optional Headers, such as VMID Device Header for the fabric
  - We are not against this, but would like to see it done in a new proposal
Thank You