Compute Express Link™ (CXL™) Link-level Integrity and Data Encryption (CXL IDE)

Raghu Makaram, Principal Engineer, Intel Corporation
David Harriman, Senior Principal Engineer, Intel Corporation
• Threat Model
• Device attestation
• CXL Integrity and Data Encryption (IDE)
  • CXL.io
  • CXL.cachemem
• Error Handling

Content based on CXL 2.0 Specification
Ack: CXL Consortium trainings from Mahesh Natu, Intel and Patrick Bailey, Microchip
Assets
- Protocol Transactions (Data and Header) sent on the link
- Cryptographic keys used to protect data on link

Adversaries and threats
- Simple or skilled hardware adversaries
- Physical attacks on the link such as use of interposers to snoop, modify, inject or replay data or headers, device swap (trusted device with untrusted one)

TCB
- IDE blocks in root-port and end-points
- CXL switch
- Agent that establishes the IDE session

Defense against denial of service is out of scope for IDE
Device Enumeration, Attestation, Key-exchange

- Device enumeration, attestation and key exchange via CXL IO
- Follow DMTF Security Protocol and Data Model (SPDM)
- PCIe IDE ECN for CXL.IO and CXL IDE Establishment ECN for CXL.cachemem
- Use endpoint Data Object Exchange (DOE) mailbox registers
Device Attestation

- Follows PCIe and SPDM
- HELLO exchange to get capabilities and negotiate parameters
- CERTIFICATE returns device certificate chain including device unique public key cert.
- CHALLENGE_AUTH: Device signs requestor provided nonce with device unique private key. Used to authenticate device
- MEASUREMENTS returns signed measurements and requestor nonce with device unique private key
IDE Key Establishment

- Follow CXL IDE establishment ECN
- Authenticated key exchange based on the trust establishment in the previous slide to establish a session
- Session used to wrap IDE keys during key programming step
- Session can be kept alive and used to refresh both session and IDE keys
### IDE Overview

<table>
<thead>
<tr>
<th>CXL.IO IDE follows PCIe IDE ECN</th>
<th>CXL.Cachemem IDE</th>
</tr>
</thead>
</table>
| • AES-GCM with 256 bit key-size for data encryption and integrity  
  • 96-bit MAC transmitted for integrity protection | • AES-GCM with 256 bit key-size for data encryption and integrity  
  • 96-bit MAC transmitted for integrity protection |
| • Link and selective streams IDE | • Link IDE only no selective streams |
| • TLPS protected by IDE  
  • TLP headers not encrypted, only integrity protected  
  • Data is encrypted and integrity protected | • All protocol flits protected by IDE  
  • 32 bits of flit header not encrypted, only integrity protected  
  • Rest of header flit and data in all-data flits encrypted and integrity protected |
| • Switches can implement or be configured for flow-through with selective streams IDE | • Switches must implement link IDE |
| • Encrypted Plaintext CRC for AES engine robustness | • Encrypted Plaintext CRC for AES engine robustness |
| • DLLP, sequence number and Link CRC not protected by IDE  
  • Control information can flow independent of IDE  
  • CRC computed on IDE content and checked before decryption/integrity check | • Control flits and Link CRC not protected by IDE  
  • Control flits can flow independent of IDE  
  • CRC computed on encrypted flit and checked before decryption/integrity check |
• New Header Slot for MAC transmission
  • H6 (110b) slot format encoding
  • Slot carries 96 bit of MAC

• New IDE control flit
  • IDE.idle: message sent as part of IDE flow to pad sequences
  • IDE.start: message to begin using programmed keys
  • IDE.TMAC: Truncated MAC send to complete MAC Epoch Early
• Keys
  • Tx and Rx keys must match
  • Key configured via SPDM session
  • Key refresh without loss of data

• Initialization Vector (IV)
  • Implicit and not transmitted over the link
  • Monotonic counter incremented for every MAC
  • Initial counter value configured as part of key exchange

<table>
<thead>
<tr>
<th>IV fields</th>
<th>Bits</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>95:92</td>
<td>Sub-stream ID = 1000b</td>
</tr>
<tr>
<td></td>
<td>91:64</td>
<td>Must be zeros</td>
</tr>
<tr>
<td></td>
<td>63:0</td>
<td>Monotonic Counter</td>
</tr>
</tbody>
</table>

- 256 bit key
- 96 bit IV

Tx - Upstream
Rx - Upstream
Tx Downstream
Rx Downstream
MAC Generation and Handling

- AES-GCM integrity value (MAC) accumulated over multiple flits
- MAC Epoch: Set of consecutive protocol flits that are aggregated for a given MAC Epoch
  - Aggregation Flit Count is the number of flits in a MAC Epoch
- Two modes of MAC aggregation and handling
  - Containment mode: MAC must be received and checked before opcode/data released out of IDE for further processing. Aggregation flit count = 5.
  - Skid mode: Opcode/data can be released for further processing without waiting for MAC. Lower latency and bandwidth impact. Requires solution stack level assessment of tradeoffs. Aggregation flit count = 128
MAC Transmission – Back-to-Back Traffic

- MAC for each MAC Epoch transmitted in order
- Typically, MAC transmitted in header slot at the earliest possible time
- No later than 6\textsuperscript{th} protocol flit after MAC Epoch ends
Early MAC Epoch Termination

- Transmitter may terminate MAC Epoch early
  - MAC transmitted as IDE.TMAC control flit
- Expected to happen as part of link idle handling
- Transmitter must send TruncationDelay number of IDE idle flits before sending protocol flits for next MAC Epoch
Encrypted Plaintext CRC (PCRC)

- Provides robustness against hard and soft errors internal to crypto engines
- Integrated into MAC check mechanism and does not consume any additional bandwidth
Error Handling

• Link errors/ retries
  • No IDE impact to link level retry mechanism
  • IDE only sees flits that pass link CRC checks

• Integrity errors on IDE
  • Error can be in data or header
  • Flits dropped and error logged/signaled
  • All subsequent IDE traffic dropped till link reset
  • Device to clear state/plain-text data or provide access control to prevent leakage of secrets

• Link reset
  • Clear secrets from device and reset IDE keys
  • Need to re-establish connection
Summary

- Device attestation and key exchange over CXL.io interface
  - SPDM and DOE mailboxes used for this purpose
- CXL.io IDE follows PCIe IDE ECN
- CXL.cachemem IDE
  - Detailed definition in CXL 2.0 specification section 11
  - Supports containment and skid modes
  - Link integrity and confidentiality in an implementation efficient manner
  - Encrypted PCRC for crypto engine robustness
  - Low bandwidth and latency impact
Q&A

Please share your questions in the Question Box
Thank You